LAMK Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

Bluetoothin hyödyntäminen automaatiossa

LAHDEN AMMATTIKORKEAKOULU
Tekniikan ala
Kone- ja tuotantotekniikka
Suunnittelupainoinen mekatroniikka
Opinnäytetyö AMK
Kesä 2016
Janne Etola

Lahden ammattikorkeakoulu
Kone- ja tuotantotekniikka

Etola, Janne:

Bluetoothin hyödyntäminen automaatiossa

Suunnittelupainoinen opinnäytetyö, 43 sivua, 4 liitesivua

Kevät 2016

TIIVISTELMÄ

Älypuhelimien ja tablettien yleistyminen markkinoilla mahdollistaa langattoman käyttöliittymän lähes jokaisen käyttöön. Bluetooth on erittäin vahva langattoman viestinnän tapa välittää tietoa, ja tämän todistaa sen laaja käyttö matkapuhelinmarkkinoilla ympärillä maailmaa.

Työn tavoitteena oli kehittää edullinen Bluetooth-pohjainen rajapinta mekatroniselle laitteelle ja pohtia Bluetoothin soveltuvuutta automaatioympäristössä signaalikaapelin korvaajana. Bluetooth standardin omaava langaton yhteys takaa nopean ja turvallisen liityntärajapinnan mekatronisille laitteille, jolloin laitetta voidaan esimerkiksi monitoroida tai konfiguroida etänä. Bluetooth-moduulin matala virran tarve sopii myös hyvin antureille, jossa langattomuus on ehdoton seikka.

Työssä perehdytään käytännön tasolla Bluetooth-yhteyden luomiseen, datan siirtoon ja mahdollisiin tietoturvariskeihin, mitä langaton viestintä voi aiheuttaa. Tutkimus perustuu käytännön kokemukseen, jonka ohella kehitettiin Bluetooth-yhteensopiva paineilmaventtiiliohjain, joka on mahdollista konfiguroida langattomasti suoraan Android älypuhelimelta.

Työn toteutus alkoi yritysideasta syksyllä 2014, jolloin oli tarkoitus kehittää idea tuotteeksi. Kehitys päättyi keväällä 2015, sillä tuotteen jatkojalostus ei ollut kannattavaa kustannussyistä.

Asiasanat: Bluetooth, paineilmaventtiili, tuoteidea, ohjelmointi

Lahti University of Applied Sciences
Degree Programme in Production and Manufacturing Technology

ETOLA, JANNE: Benefits of Bluetooth in automation applications

Bachelor's thesis in mechatronics 43 pages, 4 pages of appendices

Spring 2016

ABSTRACT

As mobile phones and tables become more and more popular for everyday use, the use of a wireless interface is possible for almost anyone. Bluetooth is a robust method to transfer data, and the proof of its success is the widespread usage in mobile phone market.

The scope of this study was to develop a low cost Bluetooth-based interface for a mechatronic device, and to reflect the feasibility of Bluetooth in automation industry as a replacement for a signal cable. Bluetooth standard guarantees a fast and safe interface for mechatronic devices and it offers the possibility of wireless configuration and monitoring. The low energy consumption of a Bluetooth module can be utilized to transfer data wirelessly from a sensor instead of using a conventional signal cable.

This study explains the creation of a Bluetooth connection, wireless data transfer, and security issues on a practical level. The study is based on personal experience how to create a Bluetooth compatible pneumatic valve controller that can be configured wirelessly using an Android smart phone.

The start of the study begun as a business idea in fall 2014, and the idea was to create an actual product. The design process ended in spring 2015 because the development of the product was not profitable in the long run.

Key words: Bluetooth, pneumatic, valve, business idea, programming

SISÄLLYS

LYHENNELISTA

ACL	Asynchronous Connection-Less
ACO	Authenticated Ciphering Offset
AES	Advanced Encryption Standard
AFH	Adaptive Frequency Hopping
BR	Basic Rate
CRA	Challenge-Response Authentication
CRC	Cyclic Redundancy Error Check
EDR	Enhanced Data Rate
EEPROM	Electronically Erasable Programmable Read-Only Memory
FCU	Fire Controller Unit
FEC	Forward Error Correction
GAP	Generic Access Profile
HCI	Host Controller Interface
HEC	Header Error Check
HS	High Speed
I/O	Input output
ISM	Industrial Scientic Medical
L2CAP	Logical Link and Adaptation Protocol
LC	Link Controller
LCD	Liquid-Crystal Display
LE	Low Energy
LM	Link manager
LMP	Link manager Protocol
MAC	Media Access Control
NFC	Near Field Communication
PWM	Pulse Width Modulation
QoS	Quality Of Service

QR	Quick Response
SCO	Synchronous Connection-Oriented
SM	Security Manager
SRES	Signal Response
SSP	Simple Secure Pairing
USB	Universal Serial Bus

1 JOHDANTO

Langaton viestintä on haaste monella sovellusalalla, sillä siihen liittyy monia ulkoisia seikkoja, jotka voivat aiheuttaa häiriöitä laitteen toimivuuteen. Haasteiden voittamiseksi Bluetooth Special Interest Group (SIG) on kehittänyt matkapuhelinmarkkinoille uuden langattoman teknologian, joka on ollut älypuhelinvalmistajien valtavassa suosiossa jo yhdeksän vuotta.

Automaatioteollisuudessa käytettävä tekniikka seuraa tietotekniikan jalan jälkiä ja tällä hetkellä se on saavuttanut langattoman Bluetooth-teknologian. Monet komponenttivalmistajat kuten ABB ja Phoenix Contact ovat julkaisseet uusia Bluetooth-yhteensopivia komponentteja, joilla laitteet voidaan modernisoida langattomiksi ja helpommiksi käyttää.

Opinnäytetyön käytännön osiossa on tutkittu, miten paineilmaventtiiliä voitaisiin ajaa käyttäen hyväksi itse rakennettua piirilevyä ja konfiguroida parametrejä käyttäen tavallista Android-älypuhelinta. Työllä ei ollut toimeksiantajaa, sillä laite oli täysin itse suunniteltu.

Työn laajuuden vuoksi sisältö on esitetty lyhyesti maallikon näkökulmasta. Työssä on kuitenkin haluttu myös korostaa tiettyjä yksityiskohtia Bluetooth-teknologiasta, millä se erottuu muista langattoman viestinnän metodeista. Opinnäytetyö on jaettu seuraaviin kappaleisiin:

- Kappaleessa 2 selvitetään Bluetoothin teknisiä hyötyjä ja rajoituksia automaatioteollisuuden näkökulmasta.
- Kappaleessa 3 perehdytään Bluetooth:in spesifikaatioon, joka on työn suurin osuus. Siinä kerrotaan Bluetooth:in eri versioiden kehityksestä ja niiden ominaisuuksista, sekä esitetty yksityiskohtia liittyen Bluetooth:in toimintaan ja tietoturvaan.

- Kappale 4 sisältää Bluetooth-topologiat, jossa kerrotaan, kuinka Bluetooth-laitteet voidaan yhdistää keskenään erilaisiin tietoverkkoihin.

- Kappaleessa 5 pohditaan Bluetooth:in sovelluskohteita automaatioteollisuudessa ja samalla esitetään esimerkkejä markkinoilla olevista laitteista, jotka käyttävät hyödyksi Bluetooth-teknologiaa.

- Kappale 6 sisältää tutkimuksen käytännönosan, jossa kerrotaan Bluetooth-yhteensopivan paineilmaventtiiliohjaimen suunnitteluvaiheista ja sen kehityksestä.

2 BLUETOOTHIN TEKNISET RATKAISUT

Bluetoothin tavoite oli saada massatuotettu langaton tiedonvälitys matkapuhelimille mahdollisimman edullisesti. Bluetooth:in suuren suosion myötä sitä on mahdollista hyödyntää myös muilla aloilla sen hyvien ominaisuuksien takia.

- Lupavapaa 2.4GHz ISM taajuus: Takaa mahdollisuuden käyttää laitetta ilman lisenssiä ympäri maailmaa. Ei häiritse Wifi-laitteita.
- Häiriönsieto: Bluetooth-teknologia mahdollistaa Adaptive Frequency Hopping (AFH)-tekniikan, jolla paritetut laitteet osaavat vaihtaa kommunikointikanavaa ilman katkoja mikäli käytetyllä kanavalla ilmenee häiriöitä.
- Matala käyttöenergia: Suunniteltu matkapuhelinkäyttöön, jolloin laite osaa itsenäisesti säästää energiaa silloin kun laite ei ole aktiivinen.
- Tietoturva: Mahdollisuus salattuun yhteyteen, mahdollisuus määrittää luotettavat laitteet.
- Audion lähetys: Mahdollisuus käyttää tiedonvälitykseen audiota digitaalisen datan sijasta.

3 BLUETOOTHIN SPESIFIKAATIO

3.1 Kehitysvaiheet

Bluetooth spesifikaatiolla tarkoitetaan standardin mukaista toimintatapaa, jonka tavoitteena on yhdistää eri valmistajien laitteet toimimaan keskenään langattomasti. Spesifikaatio takaa sen, että laitteet ovat yhteensopivia. Bluetooth-versioiden kehityksessä on otettu huomioon, että laitteet ovat rautatasolla taaksepäin yhteensopivia myös tulevaisuudessa.

3.2 Bluetooth versiot

Ensimmäiset Bluetoothin 0.8- ja 0.9-spesifikaatiot olivat salaisia, mutta spesifikaation ensimmäinen julkinen versio 1.0 julkaistiin 27.6.1991. Vuodesta 1991 eteenpäin Bluetooth on lähtenyt kehittymään vauhdilla suurten yritysten, mm. Microsoftin, 3Com:in, Lucent:in ja Motorolan tukemana. Spesifikaatiota on kehitetty jo pitkälle versiosta 1.0, joten tässä työssä tarkastellaan vain uusimpia Bluetoothin versioita 2.0; 3.0 ja 4.0 (Grandlund 2001).

3.2.1 Bluetooth 2.0 (BR/EDR)

Bluetoothin 2.0 versio Basic Rate (BR) julkaistiin vuonna 2004. Bluetooth 2.0 käyttää 2.4GHz ISM taajuutta, joka on jaettu 79 kanavaan, jotka ovat 1MHz erillään toisistaan. Taajuus on lisenssivapaa, jolloin laitteiden on luonnostaan siedettävä ympäristöstä aiheutuvaa häiriötä. Spesifikaatio tukee ACL ja SCO yhteyksiä, mikä tarkoittaa sitä että laitteet, joka on varustettu Bluetooth 2.0 moduulilla, voivat lähettää datavirtaa tai audiota. Uudempi Bluetooth 2.1 on yksi suosituimpia versioita, sillä se tukee kehittynyttä tiedonsiirtoa (Enhanced Data Rate EDR), jolloin tiedonsiirtonopeus voi olla 1-3Mb sekunnissa eli noin kolme kertaa nopeampi kuin aikaisemmilla Bluetooth-versioilla. Spesifikaation mukaan

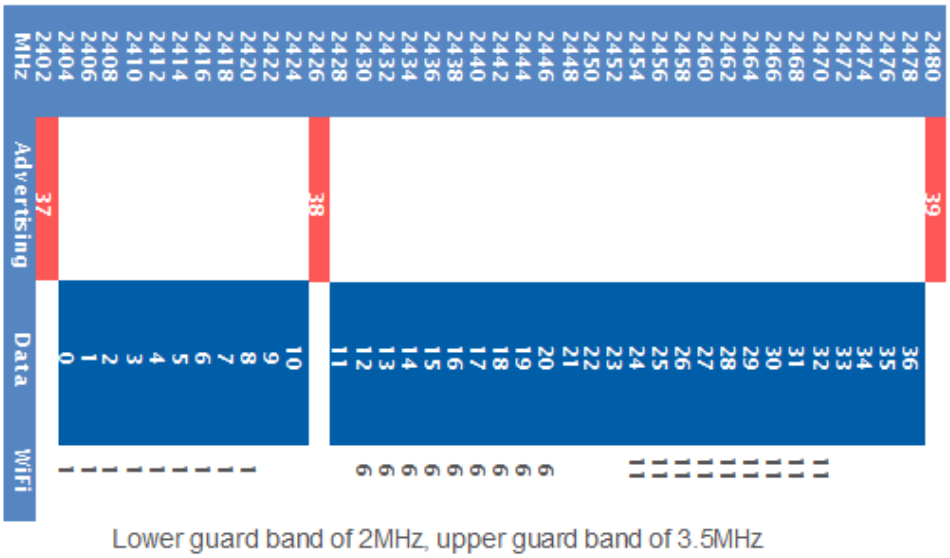
laitteen paritus on tehty helpommaksi, jolloin käyttäjän täytyy vain hyväksyä laitteen yhdistäminen. Spesifikaatio 2.1 on päivitetty turvallisemmaksi, jolloin tiedonsiirtoa on kolmannen osapuolen vaikeampi siepata. Myös virrankulutus on saatu alemmas, jolloin moduuli käyttää vain viidenneksen virtaa aikaisempiin versioihin verrattuna. Spesifikaatio tukee myös NFC-yhteyttä, jolloin laitteiden paritus nopeutuu (Bluetooth SIG, Inc 2016a.)

3.1.2 Bluetooth 3.0 (HS)

Vuonna 2009 julkaistu Bluetooth 3.0 High Speed (HS) on päivitetty versio Bluetooth 2.1:stä. Tämä käyttää täysin samaa tiedonsiirtotapaa, mutta tiedonsiirtonopeus on korkeampi, jolloin päästään jopa 24Mb/s:n nopeuksiin. Versio 3.0 käyttää hyväkseen 802.11 wifi-radiota tiedonsiirtoon ja moduulin omaa radiota vain laitteiden paritukseen ja profiilien käsittelyyn. Spesifikaatio on täysin yhteensopiva aikaisempien Bluetooth versioiden kanssa, jolloin laite voi lähettää ja vastaanottaa datapaketteja matalammalla nopeudella laitteesta riippuen (Bluetooth SIG, Inc 2016b.)

3.1.3 Bluetooth 4.0 (LE)

Bluetooth 4.0 -spesifikaatio tai toiselta nimeltään Bluetooth Low Energy (LE) tai Bluetooth Smart on suunniteltu alusta asti aivan uudella tavalla verrattuna aikaisempiin Bluetooth-versioihin. Spesifikaatio käyttää samaa 2.4GHz ISM taajuutta, mutta se on jaettu 40:een 1Mhz:ä leveään kanavaan 2MHz:n välein signaalin leveydestä johtuen. Spesifikaatio määrittää myös kolme erikoiskanavaa, jotka ovat niin sanottuja mainostuskanavia. Nämä kolme kanavaa on sijoitettu strategisesti 2.4Ghz:n spektumille, jolloin ne eivät häiritse 2.4GHz taajuudella olevia wifi-kanavia (Kuvio 1).



Kuvio 1: mainostuskanavat 2.4GHz taajuusaluella

Kanavia käytetään muun muassa löytämään muita LE-laitteita, yhteyksien aloittamiseen ja mainostamiseen. Mainostus tarkoittaa sitä, että tieto voidaan välittää ilman, että LE-laitetta on paritettu minkään muun laitteen kanssa. Tämä tarkoittaa sitä, että tieto on julkista ja se voidaan lukea millä tahansa LE-yhteensopivalla laitteella. Mainostuskanava on tarkoitettu pääsääntöisesti lähettämään vain pieniä datapaketteja. Tietoa voi myös lähettää suojatulla datakanavalla, joita löytyy yhteensä 37. Kanavat ovat 0-36 taajuudelta 2404-2424MHz ja 2428-2478MHz.

Spesifikaation päätavoite oli saada virrankulutus mahdollisimman alhaiseksi ja tehdä tiedonsiirrosta mahdollisimman yksinkertainen. Bluetooth 4.0 sopii hyvin lähettämään pieniä datapaketteja yli sadan metrin päähän riippuen lähetystehosta (Taulukko 1). Tiedonsiirtonopeus (1Mbps) on hieman alhaisempi kuin Bluetooth 3.0 ja 2.1-versiossa. Tämä ei kuitenkaan korvaa Bluetoothin aikasempia versioita, sillä tämä on tarkoitettu vai pienten datapakettien lähettämiseen hyvin pienellä energialla hyvin lyhyessä ajassa. Tieto on perillä alle kolmessa millisekunnissa, joka kattaa laitteen parituksen, tiedon välityksen ja yhteyden katkaisun (Heydon 2013a).

3.2 Bluetooth fyysinen radiorajapinta

Fyysinen radiorajapinta kattaa kaiken radioliikenteeseen liittyvän tiedonsiirron. Rajapinnalla datapaketit vastaanotetaan antennilla ja ne ohjataan eteenpäin käsiteltäviksi. Rajapinta käyttää AFH tekniikkaa välttääkseen ruuhkaisia 2.4GHz ISM kanavia. AFH tarkoittaa sitä, että jos samalla kanavalla on jokin toinen laite, lähetin ja vastaanotin vaihtavat samalle kanavalle satunnaisesti generoidun taulukon mukaan. Kanavahyppelyn ansiosta Bluetooth sietää erittäin hyvin radiohäiriöitä, sillä kanavan vaihto voi tapahtua jopa 1600 kertaa sekunnissa. Häiriötä ei tule pelkästään muista Bluetooth laitteista vaan myös ulkoisista tekijöistä, joita voivat olla esimerkiksi autojen hälyttimet ja mikroaaltouunit. AFH algorithmi osaa tehokaasti suodattaa huonoja kanavia merkitsemällä huonot kanavat taulukkoon, jolloin kanavaa ei käytetä. Bluetooth-spesifikaation mukaan laitteen on kyettävä käyttämään minimissään 20:a eri kanavaa, jolla voidaan välttää konfliktitilanteita ja pitää datalinkki mahdollisimman vahvana (Charles Hodgdon 2003). Bluetooth-linkin kantomatka riippuu radion teholuokasta (RF Wireless World 2012).

Nimi	Kantama (m)	Signaali (dBm)	Teho
Luokka 1	100	0 → +20	1 - 100 mW
Luokka 2	10	-6 → +4	250uW – 2,5mW
Luokka 3	<10	<0	< 1mW

Taulukko 1: Bluetooth-lähettimen teholuokat

3.3 Baseband

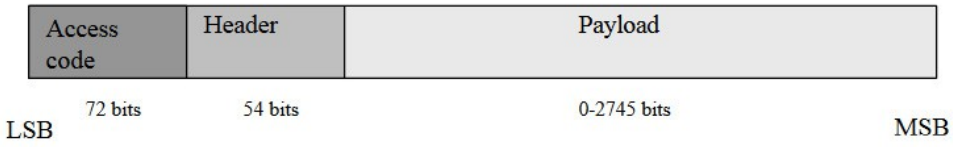
Baseband on rajapinta radioliikenteen ja Bluetooth-laitteen välillä. Rajapinta vastaa tulevien ja lähtevien pakettien ajoituksesta ja synkronoinnista käyttäen linkkitason protokollaa. Rajapinta osaa suodattaa vastaanotettavaa tiedonvirtaa muilta laitteilta. Jos suodatus

on päällä, rajapinta hylkää kaikki muut paketit, jotka eivät ole sallittujen laitteiden listalla. Tämä ei pelkästään paranna tietoturvaa vaan vähentää myös virrankulutusta.

Koska langaton radioyhteys on luonnollisesti epävarma, täytyy Baseband-rajapinnan tehdä virheenkorjausta. Virheenkorjaus tehdään käyttämällä Forward Error Correcting (FEC) -metodia lähetetyn paketin header-osassa (Galeev 2011).

3.3.1 Datapaketit

Bluetoothin tiedonsiirto on aikariippuvainen, mikä tarkoittaa sitä, että aika on jaettu 625µs:n aikaväleihin. Yksi datapaketti voi viedä yhden tai useamman paikan riippuen paketin koosta. Paketin edessä on pieni 220µs viive, jolla estetään se, että paketit eivät paketit törmää keskenään, jos dataa sattuu tulemaan samaan aikaan useasta eri laitteesta. Viive sallii myös mahdollisen kanavanvaihdon jälkeen pienen tauon, jolloin vastaanotin ehtii palautumaan. Tässä on esimerkki Bluetooth 2.0 paketista, joka koostuu kolmesta osasta (Kuvio 2) (Gelzayd 2002).



Kuvio 2: Bluetooth 2.0 datapaketti

- **Access code**: Määrittää mihin tarkoitukseen dataa käytetään. Datapaketti voidaan lähettää ilman payload:ia, jolloin voidaan esimerkiksi tiedustella dataa muilta laitteilta tai lähettää yhdistyspyynnön kantamalla olevalle laitteelle. Yhteyspyynnön lähettävä laite toimii aina master-laitteena, ja yhteyden vastaanottava laite on slave. Se määrittää myös tavallisen

tiedonsiirron kahden tai useamman Bluetooth-laitteen välillä. Access code:n sisältä generoidaan kanavahyppelyn taajuudet, jolloin laitteet pysyvät synkronissa keskenään (Liu 2016).

- **Header**: Header on osa, jota Link Controller (LC) käyttää pakettien järjestelemiseen. Header sisältää järjestysluvun ja paketin pituuden, minkä perusteella paketit voidaan järjestää oikeaan järjestykseen. Se sisältää myös virhekorjauksen (FEC) ja määrittelyn sille, halutaako tieto lähettää audiona vai data-virtana. Virheenkorjaus tarkastetaan vastaanottavassa laitteessa vertaamalla Header Error Check (HEC) -arvoja. Jotkin baseband-paketit käyttävät virheen korjaukseen Cyclic Redundancy Error Check-tarkistusta (CRC). Header on 18 bittiä pitkä, mutta virheenkorjauksen takia header:in bitit lähetetään kolmesti, jolloin kokonaispituudeksi tulee 54 bittiä (Gelzayd 2002).

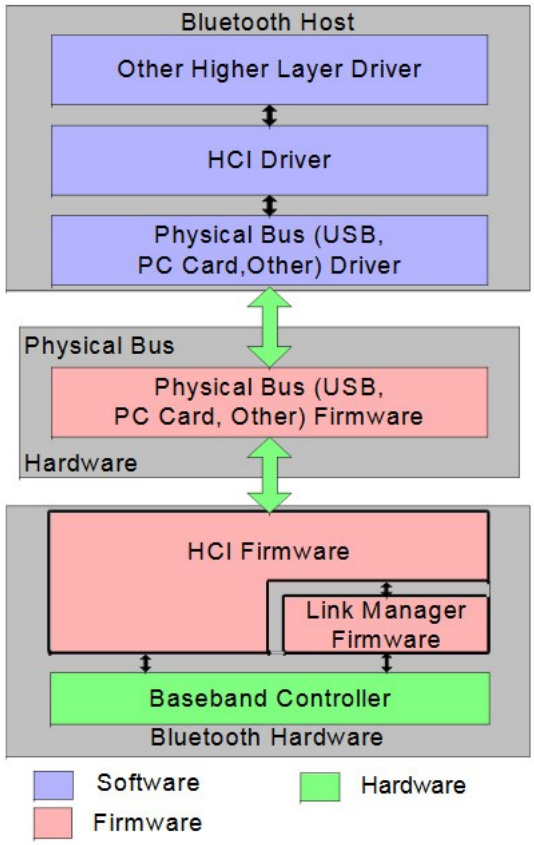
- **Payload**: Sisältää siirrettävän datan, joka voi olla ACL-tyyppistä datavirtaa tai SCO-tyyppistä audiota, jonka maksimikoko voi olla 0-2745 bittiä (Gelzayd 2002).

3.4 Link Manager Protocol (LMP)

Link manager (LM) hallitsee kaiken laitteiden välisen Bluetooth-kommunikoinnin Link Manager-protokollan (LMP) avulla. Siihen sisältyy suojatun yhteyden luominen, verkon ylläpito ja laitteiden paritus (Mäkelä 2006a).

3.5 Host Controller Interface (HCI)

HCI jakaa Bluetooth kokonaisuuden fyysiseen Bluetooth-ohjaimeen ja isäntälaitteeseen (Kuvio 3) (IEEE, NPO 1999).



Kuvio 3: Baseband ja HCI

Tämä luo komentorajapinnan LM:n ja Bluetoothin ylempien tasojen välille, jolloin pystytään antamaan käskyjä alemmalle baseband-rajapinnalle ja lukemaan sen rekistereitä ja tilatietoja. HCI-laiteohjelmisto on yhteydessä fyysisen väylän kautta ohjelmapuolen ajureihin. Yhteys voidaan muodostaa esimerkiksi USB-kaapelilla tai laitteen sisäisellä bus-väylällä. Sisäinen bus-väylä voi olla esimerkiksi älypuhelimen prosessorin ja Bluetooth-ohjaimen välinen yhteys (Bluetooth SIG, Inc 2016c).

HCI-laiteohjelmisto sisältää Host Control Transport-tason, minkä tarkoituksena on kuljettaa dataa eteenpäin ilman, että käyttäjä tuntee datan siirtoon liittyviä pikku seikkoja. Tämän tarkoitus on helpottaa

Bluetooth-ohjaimen käyttöä. HCI sisältää myös tiedon kulkuun liittyvää hallintaa, jolloin Bluetooth-ohjelma ei voi ylittää Bluetooth-ohjaimen puskuria ACL datalla, jos vastaanottava laite ei vastaa (Bluetooth SIG, Inc 2016d).

3.6 Logical Link Control and Adaption Protocol (L2CAP)

L2CAP on multipleksaus-taso, jossa paketit organisoidaan ja välitetään HCI:lle ja suoritetaan Quality Of Service (QoS) -toimenpiteitä. Koska basebandin lähettävän paketin koko on rajattu 2745 bittiin, täytyy L2CAP rajapinnan tehdä resurssin hallintaa, ennen kuin paketit voidaan lähettää vastaanottavalle laitteelle. Paketit eivät voi myöskään voi ylittää HCI-kanavan siirtonopeutta. L2CAP voi vastaanottaa maksimissaan 64kB paketteja, jotka se paloittelee helpommin käsiteltäviksi. Paloitellut paketit kootaan uudestaan baseband:in puskurille sopiviksi ja lähetetään basebandille.

L2CAP-tasolla on mahdollista suorittaa virheenkorjausta, jolla voidaan suodattaa huomaamatta läpi päässeet virheet (Bluetooth SIG, Inc 2016c).

3.7 Security Manager (SM)

Security Manager hallitsee ja tallettaa salatun yhteyden muodostukseen tarvittavia avaimia sekä vastaa tietoturvaan liittyviin pyyntöihin. Se pitää listaa luotettavista laitteista, ja huolehtii datan enkryptaamisesta sekä dekryptaamisesta (Padgette, Scarfone, Chen 2012).

3.8 Generic Access Protocol (GAP)

GAP määrittää toimenpiteet laitteiden paritukselle, suorittaa salaisen tiedon välityksen ja yhteyden muodostamisen. Profiilin ansiosta jokaisen

laitteen ei tarvitse kehittää omaa tapaa ottaa yhteyttä muihin laitteisiin, vaan ne voivat käyttää profiilia suorittaakseen yhteydenmuodostukseen tarvittavat toimenpiteet (Heydon 2013a).

3.8 Tietoturva

Tietoturva on langattoman yhteyden yksi tärkeimmistä seikoista. Hyvällä tietoturvalla estetään arkaluontoisen tiedon vuotaminen ulkopuolisten käsiin ja tiedon salakuuntelu. Kun yhteys on salattu salausavaimella, yhteys näyttää ulkopuolisen silmissä satunnaiselta datalta, eikä sitä voida lukea ilman salauksen purkamista salausavaimella. Tietoturvatoimenpiteet estävät myös ulkopuolisen tiedon pääsemistä vastaanottimelle, jolloin kolmas osapuoli ei voi käyttää Bluetooth-laitetta ollenkaan. Tiedon pääsy voidaan estää esimerkiksi laatimalla lista sallittujen laitteiden MAC-osoitteesta ja sallia ainoastaan tunnetut laitteet. Kolmas tietoturvariski on radiohäirintä, jolloin kolmas osapuoli haluaa tahallisesti estää kaiken tiedonsiirron tiedon laadusta välittämättä. Tämä tarkoittaisi sitä, että suuritehoisella lähettimellä lähetetään Bluetooth-vastaanottimeen satunnaista dataa, joka peittää hyötydatan pääsyn vastaanottimelle. Radiohäirinnältä ei voida suojautua muuten kuin pitämällä Bluetooth-laitteiden väliset lähetystehot matalina tai eristämällä ympäristö 2,4GHz radiotaajuudelta. Näin hyökkääjä ei löydä laitetta, johon hyökätä. Tahallinen radiohäirintä on laitonta Suomen laissa sekä suurimmassa osaa muualla maailmaa (FINLEX 2007).

3.8.1 Tietoturvatasot

Bluetoothin tietoturva perustuu laitteiden väliseen salattuun tiedonsiirtoon, ja Bluetoothin spesifikaatio määrittää neljä eri tietoturvatasoa:

1. Suojaamaton taso: Laite ei suorita tietoturvaan liittyviä toimenpiteitä eikä myöskään suojaa dataa salausavaimella.

2. Service-level enforced security: Laitteet voivat muodostaa suojaamattoman ACL-yhteyden, ja suojaustoimenpiteet aloitetaan vasta L2CAP-tasolla. Bluetooth-ohjaimen Security Manager-taso hallitsee yhteyksien turvallisuutta estämällä epäluotettavat MAC-osoitteet. Suojaustoimenpiteet tehdään vasta kun pyyntö salatun yhteyden muodostamiseen on tehty. Suojattu yhteys on vapaavalintainen.

3. Link level enforced security: Tietoturvatoimenpiteet alkavat linkkitasolla. Yhteyttä muodostavan laitteen täytyy saada hyväksyntä, ennen kuin se pääsee ottamaan yhteyden L2CAP-tason jälkeisiin palveluihin.

4. Link level enforced security with encrypted key exchange: Tämä suojaustaso on vain ainostaan Bluetooth-version 2.1 + EDR jälkeisiin Bluetooth-ohjaimiin. Se on vahvistettu suojaustaso, joka on vastaavanlainen kuin tietoturvataso kaksi, mutta se tukee Simple Secure Pairing (SSP) -ominaisuutta.

3.8.2 Turvallisen yhteyden muodostus

Kun Bluetooth-laite haluaa ottaa yhteyden toiseen Bluetooth-laitteeseen, salatun yhteyden luomiseksi täytyy yhteyden muodostavan laitteen tietää vastaanottavan laitteen paritusavain. Paritusavaimen jakamisella voidaan varmistaa, että laite on luotettava. Bluetooth-versiot 2.0+EDR asti perustuvat täysin paritusavaimen jakamiseen, jossa luodaan useampi 128-bittinen avain salausta varten. Salaus on nimeltään E0-salaus, joka käyttää hyödykseen satunnaislukuja salaisen avaimen muodostamiseen.

Salaus aloitetaan luomalla molemmille laitteille sama *Initialization key* (K_{init}), kun laitteet ottavat yhteyden ensimmäistä kertaa (Kuvio 4). K_{init}

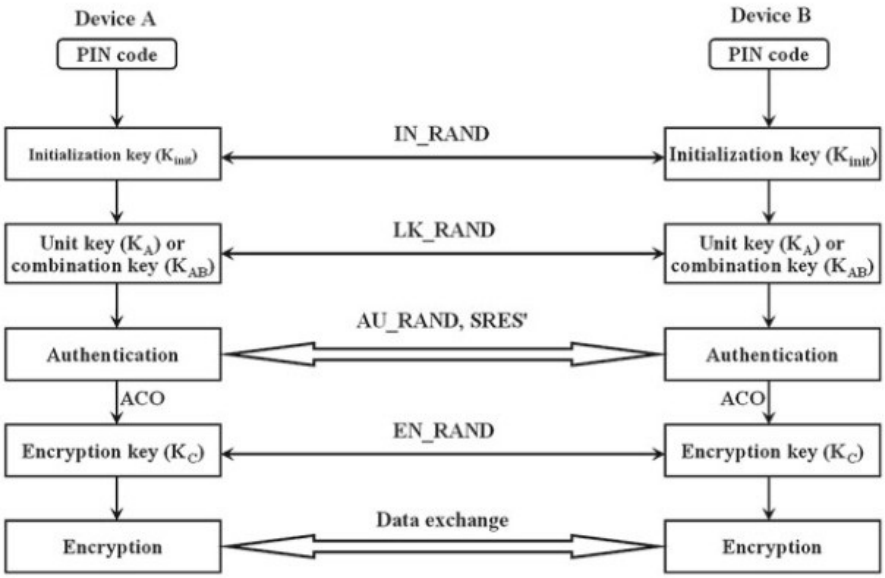
avainta käytetään seuraavan neljän avaimen luomiseen. K_{init} luodaan paritusavaimesta, IN_RAND satunnaisluvusta ja laitteen omasta uniikista BD_ADDR osoitteesta. Täytyy kuitenkin huomata, että IN_RAND satunnaisluku kulkee käsittelyn aikana salaamattomana.

Unit Key (K_A) on avain, joka luodaan laitteesta A ja enkryptataan se avaimella K_{init}. Tämän jälkeen avain K_A lähetetään laitteelle B, minkä jälkeen se enkryptaa avaimen K_A avaimella K_{init}. Näin laite B saa itselleen salaisen avaimen K_A. Kun avain K_A on luotu, se voidaan jakaa kaikille luotettaville laitteille, jolloin ne voivat kuunnella yhteyttä. Avaimen käyttö ei kuitenkaan ole kaikista varmin tapa suojautua vakoilulta, eikä Bluetooth-spesifikaatio ei suosittele avaimen käyttöä ainoana tietosuojamenetelmänä.

Combination Key (K_{AB}) luodaan samalla tapaa kuin K_A, mutta siinä käytetään laitteen B generoimaa (K_B) avainta. Nämä avaimet yhdessä muodostavat avaimen K_{AB}.

Seuraavaa vaihetta kutsutaan Challenge-Response Authentication:ksi (CRA), jossa tarkastetaan tietääkö laite oikean paritusavaimen. Prosessin aikana lähetetään uusi salaamaton 128-bittinen AU_RAND-avain ilmassa laitteelle A. Tuloksena kummatkin laitteet muodostavat uuden 32-bittinen Signal Response (SRES) -avaimen ja 96-bittinen ACO-avaimen (Authenticated Ciphering Offset). Laitteet vertaavat SRES-avaimia keskenään, ja jos avaimet täsmäävät, laiteet ovat yhteydessä. ACO-avainta käytetään salatun yhteyden salausta varten seuraavassa vaiheessa.

Encryption Key (K_c) generoidaan molemmille laitteille avaimista ACO ja K_A tai K_{AB} ja satunnaisluvusta EN_RAND, mikä lähetetään salaamattomana laitteelta A laitteelle B. Tämän jälkeen salattu yhteys on valmis käytettäväksi, jolloin molemmat laitteet voivat enkryptata ja dekryptata viestin K_c-avaimella ja lukea sen sisällön (Haataja, Hyppönen, Pasanen, Toivanen 2013).



Kuvio 4: Salauksen muodostus

3.8.3 Tietoturvariskit

Koska salauksen luomisessa käytetään usein neljä merkkistä paritusavainta, suojaus ei ole välttämättä vahvin, koska avaimen voi pakottaa auki arvaamalla avaimen monta kertaa. Tämä voi johtaa niin sanottuun man-in-the-middle hyökkäykseen, jossa henkilö kuuntelee ja muokkaa lähetettäviä viestejä. Tämän takia Bluetooth-versio 2.0+EDR ei ole luotettavin tapa jakaa tietoa. Tietoturvaa on parannettu Bluetooth 2.1-versiossa, joka tarjoaa SSP-ominaisuuden. Bluetooth 4.0 puolestaan käyttää aivan uutta tietoturvaprotokollaa nimeltään AES-CCM (Bluetooth SIG, Inc 2016e). AES-CCM käyttää kuitenkin samaa paritusavain-metodia kuin E0-salaus, mikä voi mahdollistaa salakuuntelun. Jos hyökkääjä saa käsiinsä salaamattomat satunnaisluvut parituksen yhteydessä kuuntelemalla salaamatonta langatonta yhteyttä, niin tämä voi johtaa tietoturvariskeihin (Haataja, Hyppönen, Pasanen, Toivanen 2013). Jos laite on paritettu turvallisessa paikassa matalalla lähetysteholla, riski salatun paritusavaimen vuotamiseen on erittäin pieni.

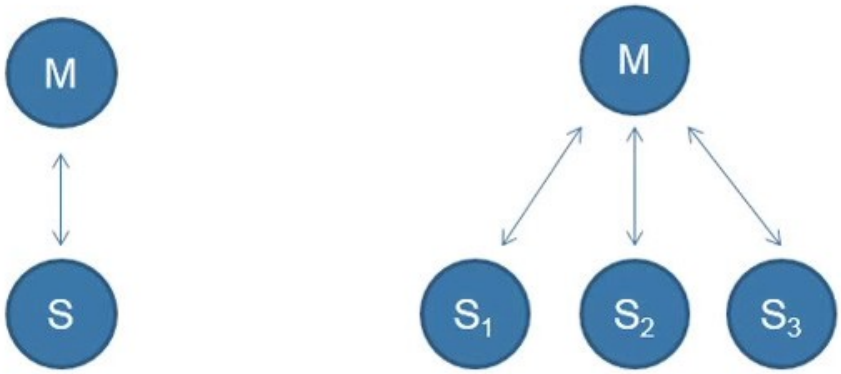
Tietoturvaa voidaan kasvattaa ohjelmatasolla siltä varalta, jos paritusavain pääsee vääriin käsiin. Bluetooth-ohjelma voi kysyä käyttäjältä käyttäjätunnuksia, minkä jälkeen laite on valmis käytettäväksi. Tiedonsiirto voidaan ohjelmatasolla enkryptata vielä uudelleen määriajoin vaihtuvalla avaimella, jolloin tietoturva paranee entisestään.

Tietoturvaa ei välttämättä tarvita esimerkiksi anturi-tasolla, jos anturin tehtävä on lähettää passiivisesti dataa. Tällöin vasteaikaa voidaan pienentää, sillä Bluetooth-ohjaimien ei tarvitse salata viestiä ollenkaan.

4 BLUETOOTH TOPOLOGIAT

4.1 Piconet

Bluetooth-laite on mahdollista yhdistää moneen eri Bluetooth-yhteensopivaan laitteeseen, jolloin laitteet muodostavat verkon, jota kutsutaan piconetiksi (Kuvio 5). Piconet-laitteet toimivat aina samalla taajuudella, ja laitteet tunnistetaan toisistaan uniikilla 48-bittisellä osoitteella, joka on kirjoitettu Bluetooth-ohjaimen muistiin. Bluetooth BR:n ja EDR:n muodostama verkko tukee ainoastaan maksimissaan seitsemää aktiivista slave-laittetta yhden master-laitteen kanssa. Bluetooth low energy-laitteiden verkko puolestaan tukee yli kahta biljoonaa laitetta. Slave-laitteet eivät voi olla yhteydessä keskenään, koska master-laite hallitsee tiedonkulkua, synkronointia ja määrää mikä laite on puheenvuorossa. Tiedon on aina kuljettava master-laitteen läpi. Master- ja slave-laitteet voivat olla fyysisesti samanlaisia, mutta roolit muodostuvat yhteyden muodostuksessa. Jos laite odottaa yhteyden muodostusta, siitä tulee slave ja jos laite etsii muita laitteita, siitä tulee master (Heydon 2013b).

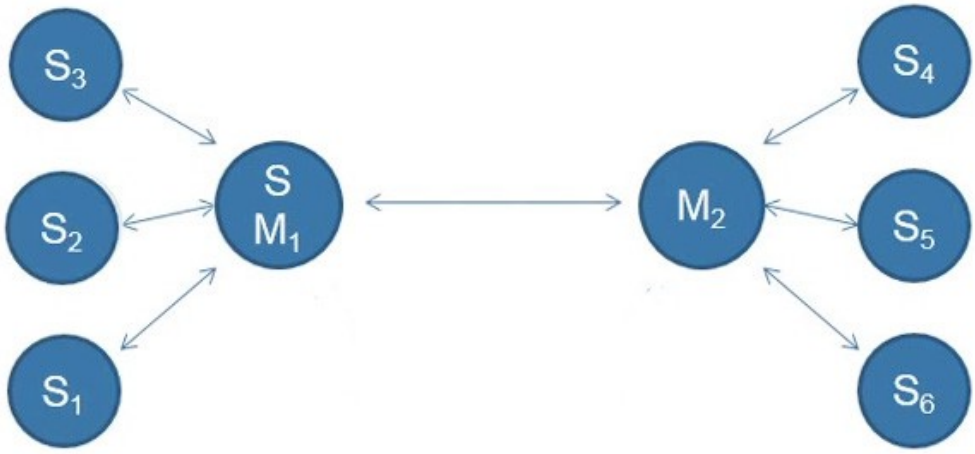


Kuvio 5: Piconet

Yhteys muodostetaan Ad-Hoc-tekniikan avulla, joka sallii laitteelle väliaikaisen yhteyden, minkä jälkeen se on vapaa irtautumaan verkosta milloin vain (Pushpa 2007).

4.2 Scatternet

Scatternet muodostuu kun kaksi piconettiä yhdistyvät (Kuvio 6). Tällä tavalla kaksi piconet:iä voi keskustella keskenään ja jakaa dataa keskenään. Piconetit käyttävät omaa AFH-taajuuksia, jolloin ne eivät häiritse toisiaan. Tämä estää myös scatternet:in lamautumisen täysin, jos toisen piconetin datan kulkuun tulee esteitä. Kun master-laite M2 ottaa yhteyden toiseen master-laitteeseen M1, M1 muuttuu M2-laitteen slave-laitteeksi. Slave-laitteeksi muuttunut master-laite M1 omistaa edelleen sen omat slave-laitteet S1, S2 ja S3. Master-laite M1 voi lähettää dataa S1-, S2- ja S3-laitteille mutta ei S4-, S5- ja S6-laitteille. Slave-laitteeksi nimitetyt laitteet eivät voi edelleenkään keskustella keskenään oma-aloitteisesti. Jotta tieto kulkisi slave-laitteelta slave-laitteelle verkon yli, scatternet:in master-laitteen M2 täytyy pyytää sen slave-laitteelta M1 pyyntö saada dataa esimerkiksi slave-laitteelta S2. Tieto slave-laitteelta S2 siirtyy sen master-laitteelle M1, josta tieto kulkeutuu scatternet:in master-laitteella M2, josta data voidaan lähettää esimerkiksi slave-laitteelle S6 (Heydon 2013b).



Kuvio 6: Scatternet

5 SOVELLUSKOHTEET AUTOMAATIOSSA

5.1 Automaatioteollisuuden uudet sovelluskohteet

Automaatioteollisuudessa laitteet ovat kiinteitä ratkaisuja, joita ohjataan painonapein tai mahdollisesti resistiivisen kosketusnäytön kautta. Tämä voi rajoittaa tiettyjen sovelluskohteita, jossa vaaditaan langattomuutta esimerkiksi korkeat nostokurjet tai robotin työkalu. Yksinkertaisen Bluetooth-rajapinnan kautta kiinteä laite voidaan yhdistää esimerkiksi älypuhelimeen, tablettiin tai kannettavaan tietokoneeseen. Edullisten Bluetooth-moduulien integrointi komponentteihin mahdollistaa esimerkiksi komponentin monitoroinnin ja konfiguroinnin. Langaton viestintä mahdollistaa langattoman anturoinnin, jossa tilatiedot voidaan välittää ilmateitse satojen metrien päähän riippuen lähettimen teholuokasta. Tietoa voidaan myös välittää muille laitteille, jolloin kenttälaite pystyy havaitsemaan ympärillään olevia laitteita.

5.2 Monitorointi ja konfigurointi

Kenttälaitetta voidaan monitoroida ja konfiguroida uusilla arvoilla parittamalla konfigurointilaite kommunikoimaan kenttälaitteen kanssa. Bluetooth takaa kaksisuuntaisen katkeamattoman tiedonsiirtomahdollisuuden, jolla se voi vastaanottaa tietoa esimerkiksi lämpötilaa, painetta ja tai sekvenssin asemaa. Tämä mahdollistaa esimerkiksi dataloggerin käytön, jolloin laitteen antamat arvot voidaan tallentaa langattomasti SD-kortille. Konfiguroinnin yhteydessä kenttälaitteelle voidaan antaa käskyjä Bluetoothin välityksellä, jolloin sitä voi käyttää myös ohjauspaneelina. Kenttälaite voi sijaita missä tahansa käden ulottumattomissa, mutta silti Bluetooth moduulin kantamalla. Tiedonsiirto tehdään syöttämällä merkkijono konfigurointilaitteelta

Bluetoothin yli kenttälaitteen Bluetooth-moduulille, minkä jälkeen arvot parsitaan merkkijonosta oikeille muistipaikoille. Konfigurointi voidaan toteuttaa graafisen käyttäjäystävällisen käyttöliittymän kautta, jolloin konfigurointi onnistuu helposti esimerkiksi älypuhelimelta.

5.3 Antureiden tiedonsiirto

Anturit voidaan varustaa Bluetooth mobuulilla, ja kytkeä toimimaan kenttälaitteen Bluetooth moduulin kanssa. Langattoman tiedonsiirron ansioista anturit voidaan sijoittaa hankaliin paikkoihin, jossa kaapelointi olisi mahdotonta toteuttaa. Antureissa voidaan hyödyntää joko jatkuvaa tiedonsiirtoa, jolloin anturi on jatkuvasti yhteydessä isäntälaitteessa tai purskemaista tiedonsiirtoa, joka on nopeampi ja kuluttaa vähemmän energiaa, mutta mahdollistaa pienemmän datan käytön. Jatkuvassa yhteydessä olevan anturin alhaisen tiedonsiirtonopeuden takia anturilta ei voida vastaanottaa nopeisiin ajastuksiin liittyvää dataa, sillä viive anturin aktivoitumiseen ja tiedon vastaanottamiseen voi olla satoja millisekunteja. Purkemainen tieto on mahdollista välittää Bluetooth LE-yhteensopivalta anturilta kenttälaitteelle alle kolmessa millisekunnissa (Robin Heydon 2013c). Jotta anturista saadaan täysin langaton, voidaan se varustaa paristolla tai käyttää syödyksi ympäristöstä vapautuvaa loisenergiaa.

5.4 Kenttälaitteiden keskeinen kommunikointi

Langattomalla yhteydellä voidaan vähentää kaapelointia ja nopeuttaa tuotantolinjojen asennusta, jossa modulaarisuus on tärkeää. Kaksi täysin erillaista kenttälaitetta voi kommunikoida keskenään välittäen dataa Bluetooth-moduulien kautta. Data voi olla esimerkiksi sekvenssin tilaa, tuotantokappaleeseen liittyvää dataa tai voidaan varoittaa laitteita mahdollisesta vaarasta esimerkiksi trukin läsnäolosta, jolloin tuotantolinja

pysähtyy. Vihivaunu voidaan varustaa Bluetooth-yhteen sopivaksi, jolloin se osaa varoa muita vihivaunuja aistimalla heikkotehoista langatonta signaalia, jota muut vihivaunut lähettävät. Jokaisella vaunulla on oma tunnus, jolla vaunut tunnistavat toisensa. Vanhat teollisuuslaitteet voidaan modernisoida langattomaksi helposti asentamalla Bluetooth-ohjaimet sähkökeskuksiin.

5.4 Referenssikohteet

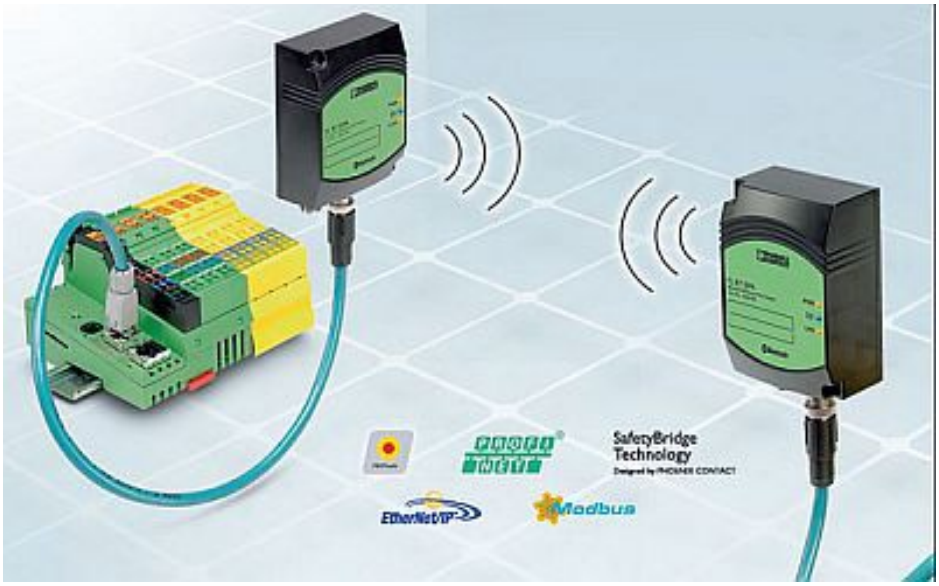
Bluetooth I/O -rajapinta

Suuri komponenttivalmistaja Phoenix Contact tarjoaa langattoman I/O-rajapinnan Bluetooth-yhteyden ylitse (Kuvio 7). Järjestelmä voi lähettää 16 eri digitaalista signaalia ja kahta analogista signaalia molempiin suuntiin samanaikaisesti. Tämä tarkoittaa, että järjestelmä voi korvata 40 signaalin kaapelin. Langatonta I/O-rajapintaa voidaan hyödyntää esimerkiksi 6-akselisen robotin työkalussa, jolloin ei tarvitse tehdä erillisiä johdotuksia robotin rakenteeseen.

Kuvio 7: Bluetooth I/O

Bluetooth Ethernet

Toinen Phoenix Contactin tuote on langaton yhteys kahden ethernet-laitteen välille (Kuvio 8). Näin Ethernet-yhteyden voi viedä langattomasti laitteelta laitteelle jopa yli satojen metrien päähän. Asennus tapahtuu asentamalla lähetin-vastaanottimet lähelle toimilaitteita ja kytkemällä ne Ethernet-kaapelilla laitteisiin. Lähetin-vastaanottimet osaavat pariutua automaattisesti nappia painamalla (Seltec Automation LLP 2015).



Kuvio 8: Bluetooth Ethernet

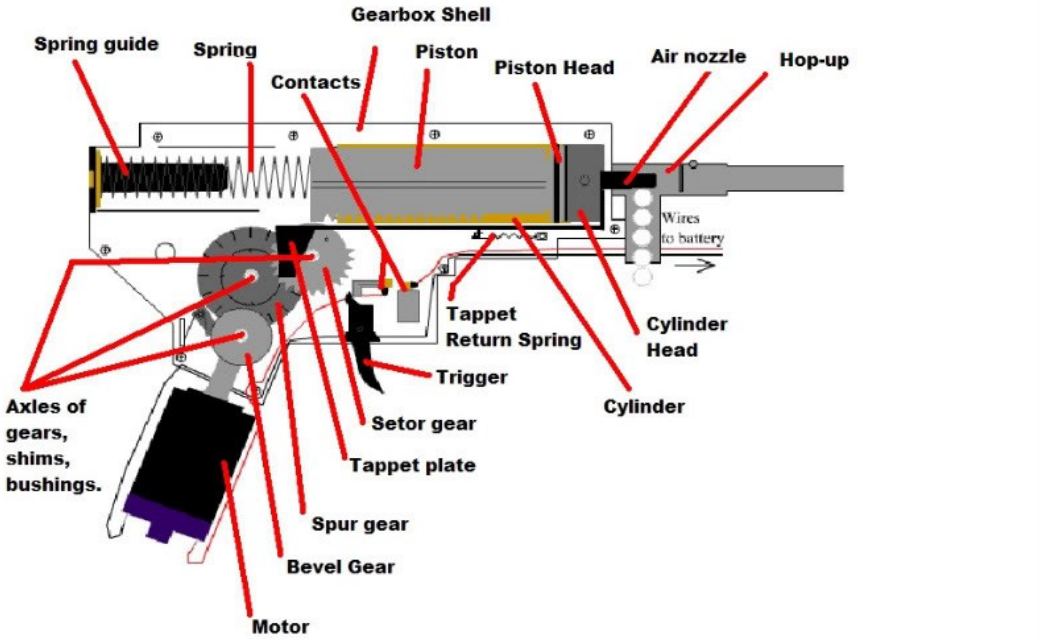
6 OMA CASE

6.1 Taustatietoa

Syksyllä 2014 tarkoitus oli kehittää yhteistyöllä Bluetooth-yhteensopiva paineilmakäyttöinen muovikuulapyssyn koneisto. Tarkoitus oli valmistaa tuote valmiiksi asti ja myydä sitä mahdollisesti eteenpäin toiminimen alla. Sen ideana oli korvata pyssyn mekaaniset osat koneistetulla alumiinipalalla, jossa oli kiinni paineilmaventtiili ja suutin. Muutoksen ansiosta muovikuulapyssy saa energiansa paintball-paineilmasäiliöstä sen sijaan, että se käyttäisi energian kehittämiseen sähkömoottoria. Pääsyynä, miksi valittiin paineilma oli se, että mekaaniset osat eivät tuottaneet tasaista lähtönopeutta ja koneiston ajamisessa syntyi tarpeettomia viiveitä. Toinen syy oli se, että mekaaniset osat eivät kestä suuria voimia, ja laadukkaat mekaaniset osat ovat kalliita ja vaikea asentaa oikein. Mekaaninen koneisto on halpa massatuottaa Kiinan-markkinoilla, mutta se on rakenteeltaan heikko materiaalivalinnan takia ja hidas toimimaan. Paineilma-koneisto on kustannuksiltaan kalliimpi kuin mekaaninen, mutta se tarjoaa ominaisuuksia, joita on lähes mahdoton toteuttaa mekaanisella koneistolla. Tämä on suuri syy, miksi paineilmakoneistojen suosio on kasvanut maailmalla ja uusia koneistoja kehitetään jatkuvasti. Muiden valmistajien hintakilpailun takia tuotteen kehitys loppui sillä sen kehitys, ja valmistus olisi tullut maksamaan liikaa. Tuote saatiin kehitettyä prototyyppitasolle, ja tuotteen kehitys oli tuottoisaa oppimisen kannalta, ja siinä opittiin hyödyntämään sulautuvaa järjestelmää mekatroniikan tukena.

6.1.1 Mekaanisen koneiston toiminta

Mekaaninen koneisto toimii siten, että liipaisin liikkeen jälkeen sähkömoottori alkaa pyörittää rattaita, minkä seurauksena mäntä vetäytyy taakse (Kuvio 9). Mäntä on yhteydessä sisällä olevaa jouseen, joka virittyy



Kuvio 9: Mekaaninen muovikuulapyssyn koneisto

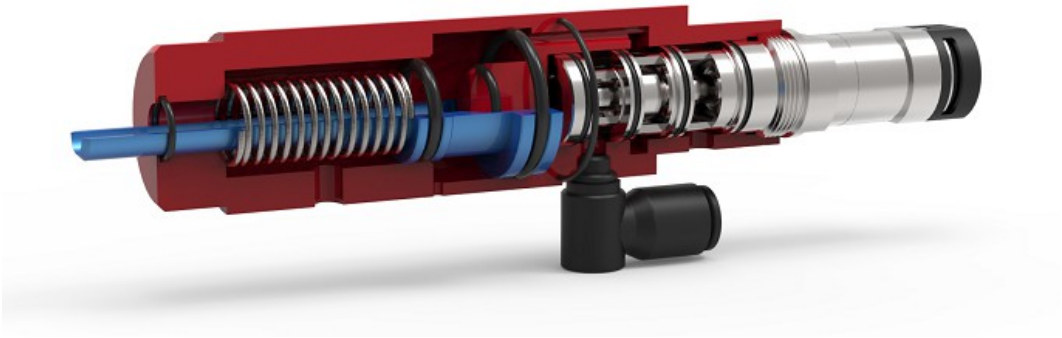
mäntää vedettäessä taakse. Kun ylin sektoriratas on päässyt pyörähtämään tiettyyn asentoon, se päästää männän liikkumaan eteenpäin, jolloin jousen energia painaa männän sylinterin sisällä sylinterin päätä vasten. Tällöin sylinterissä syntyy ylipaine, josta ilma ohjataan suuttimen kautta piippuun. Piipussa oleva kuula saa voiman ylipaineesta, jolloin se pääsee liikkumaan piipussa eteenpäin. Kuulan saama teho on reilun 1 joulen luokkaa, mikä vastaa luunappia osuessaan kohteeseen noin 40 metrin päässä.

6.1.2 Paineilmakoneiston toiminta

Energianlähteenä muovikuulapyssy käyttää paintball-paineilmatankkia, josta ilma on reguloitu 120psi:n luokkaan riippuen halutusta lähtönopeudesta. Koneiston ilmankulutus on saatu erittäin pieneksi, jolloin litran kokoisesta tankista saadaan irti tuhansia syklikertoja riippuen asetuksista.

Yksi syklikerta toimii siten, että ilma johdetaan tankista letkua pitkin muovikuulapyssyn pikaliittimelle, joka on kiinni koneistetussa alumiinipalassa. Alumiinipala sisältää ilmatiet suuttimelle ja kiinnityksen

venttiilille, mikä ohjaa ilman kulkua. Liipaisinliikkeen jälkeen paineilmaventtiili aukeaa noin kymmeneksi millisekunniksi. Tämä luo ylipaineen tankin ja piipun välille. Ilma tulee letkua pitkin venttiilin läpi suuttimelle, joka työntyy eteenpäin tehden latausliikkeen ja ohjaten kuulan piippuun. Paine työntää kuulan ulos piipusta, minkä jälkeen venttiili suljetaan ja koneisto odottaa noin 40 millisekuntia ennen seuraavaa sykliä. Suutin vetäytyy taakse joko jousen voimalla tai paineilmalla riippuen mallista.



Kuvio 10: Nemesis Engine paineilmakoneisto

6.1.3 Paineilmakoneiston edut

Paineilmakoneisto ei vaadi erityistä huoltoa muuta kuin o-renkaiden rasvausta ja se kestää helposti satojatuhansia syklejä ennen koneiston avaamista ja o-renkaiden vaihtoa. Koneiston etuna on se, että sen sisällä on ainoastaan yksi liikkuva osa, joka ohjaa piippuun ilmaa ja tekee latausliikkeen (Kuvio 10). Tällöin voidaan eliminoida syklin tolerassivirheet, jotka vaikuttavat ilman kulun tasaisuuteen. Koneisto laukaisee ammuksen alle kymmenessä millisekunneissa erittäin tasaisella lähtönopeudella, jolloin tehokas kantama on aivat eri luokkaa verrattuna mekaaniseen koneistoon. Mekaaninen puolestaan joutuu pyöräyttämään ratastoa sähkömoottorin avulla, ja tähän voi mennä helposti yli 100 millisekuntia. Mekaanisen koneiston viritys vaatii paljon taitoa ja kokemusta alalta, kun

taas paineilmakoneiston asennuksen saaminen samalle tasolle on helppoa. Suuren kysynnän takia paineilmakoneistojen hinnat ovat tällä hetkellä korkeat, mutta hinnat tippuvat, mitä enemmän uusia tuotteita tulee markkinoille.

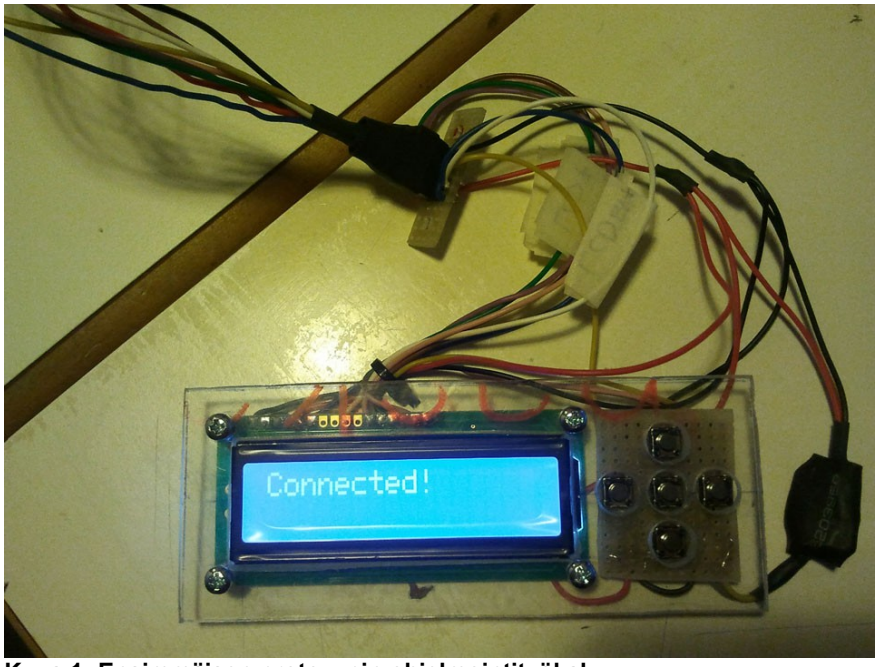
6.2 Bluetooth-ohjaimen kehitysvaiheet

Paineilmaventtiilin ohjain on pieni piirilevy, joka ohjaa paineilmakoneiston kaikkia toimenpiteitä. Tässä työssä sille on annettu nimeksi FCU, mikä tarkoittaa Fire Controller Unit:ia. Sen tarkoitus on huolehtia venttiilin ajastuksesta ja ohjata virtaa akulta venttiilille. Tavoitteena oli tehdä ensimmäinen massatuotettu Bluetooth-FCU, jota pystyy konfiguroimaan suoraan älypuhelimelta ilman että käyttäjän tarvitsee avata pyssyn runkoa.

Koneisto on suunniteltu istumaan suoraan muovikuulapyssyn rataslaatikkoon, jolloin asennus on erittäin helppoa. Pyssyn alkuperäinen mekaaninen liipaisinyksikkö korvataan piirilevyllä, jossa on mikrokytkin ja johdot FCU:lle. FCU on varustettu kaupallisella HC-05 Bluetooth 2.0+EDR-moduulilla, joka on yhteydessä piirilevyn mikrokontrollerille. Älypuhelimelta lähetetyt asetukset tallentuvat suoraan FCU:n EEPROM flash-muistiin. Asetukset pysyvät voimassa myös sen jälkeen, kun akku irrotetaan FCU:sta.

6.4.1 Piirilevyn prototyypit

Ensimmäinen FCU-versio oli tehty toimimaan LCD-konfigurointityökalulla (Kuva 1), mikä kiinnitettiin FCU:n 10-pinnisellä kaapelilla. Konfigurointityökalu sisälsi taustavalaistun LCD-näytön,



Kuva 1: Ensimmäisen protoyypin ohjelmointityökalu

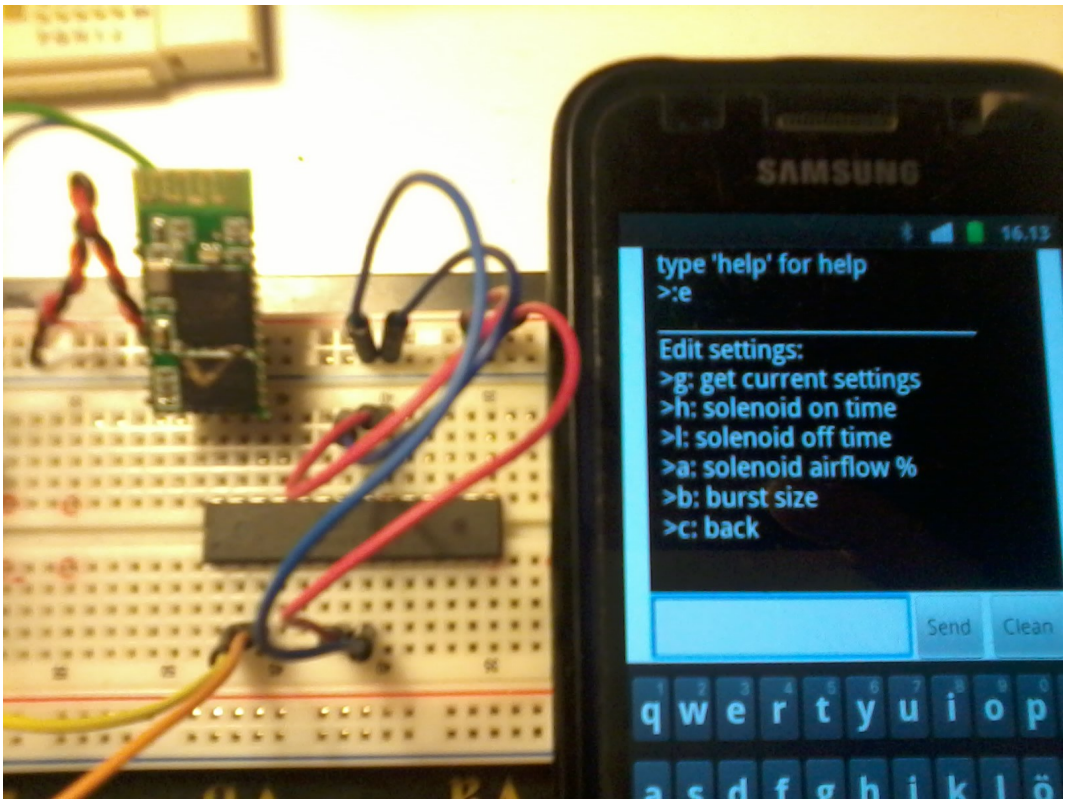
viisi painonappia ja I/O:t FCU:lle. Mikrokontrollerina käytettiin Atmega328-mikrokontrolleria, mikä oli kolvattu käsin prototyyppilevylle. Ohjelmointityökalu oli melko kömpelö ratkaisu, mutta se toimi käytännöntasolla moitteettomasti. Tämä oli kuitenkin huono ratkaisu, sillä ohjelmointityökalu olisi pitänyt suunnitella massatuotantoon, ja tämä olisi maksanut paljon. Myös LCD-näyttö vaati enemmän I/O-pinnejä ja tämän takia suuremman mikrokontrollerin. Tilanpuutteen takia ei voitu käyttää Atmega328-mikrokontrolleria, sillä FCU täytyy saada mahtumaan erittäin pieneen tilaan muovikuulapyssyn rungon sisällä.

Keskustelimme langattoman yhteyden mahdollisuudesta, jolloin puheeksi tuli Bluetooth. Aluksi idea kuulosti haastavalta, mutta muutaman tunnin internet hakutulosten jälkeen löydettiin ratkaisu. HC-05 oli 3.3V-logiikalle tarkoitettu kaupallinen Bluetooth 2.0+EDR moduuli, jota pystyy käyttämään Atmelin mikrokontrollerin kanssa. Yhteys luotiin yksinkertaisesti yhdistämällä moduulin Tx ja Rx-pinnit mikrokontrollerin serial portin kautta. Tämä tarkoitti sitä, että pystyttiin käyttämään 8-bittistä Attiny85 mikrokontrolleria, jossa on viisi kappaletta I/O-pinniä, josta kaksi menevät Bluetooth-moduulille, yksi liipaisimen mikrokytkimelle, yksi

tulenvalitsimen mikrokytkimelle ja yksi paineilmaventtiilin teho-fetille. Atiny85 on erittäin pieni pintaliitoskomponentti ja täydellinen komponentti ohjaamaan FCU:n yksinkertaisia toimintoja.

6.4.2 Käyttöliittymän kehitys

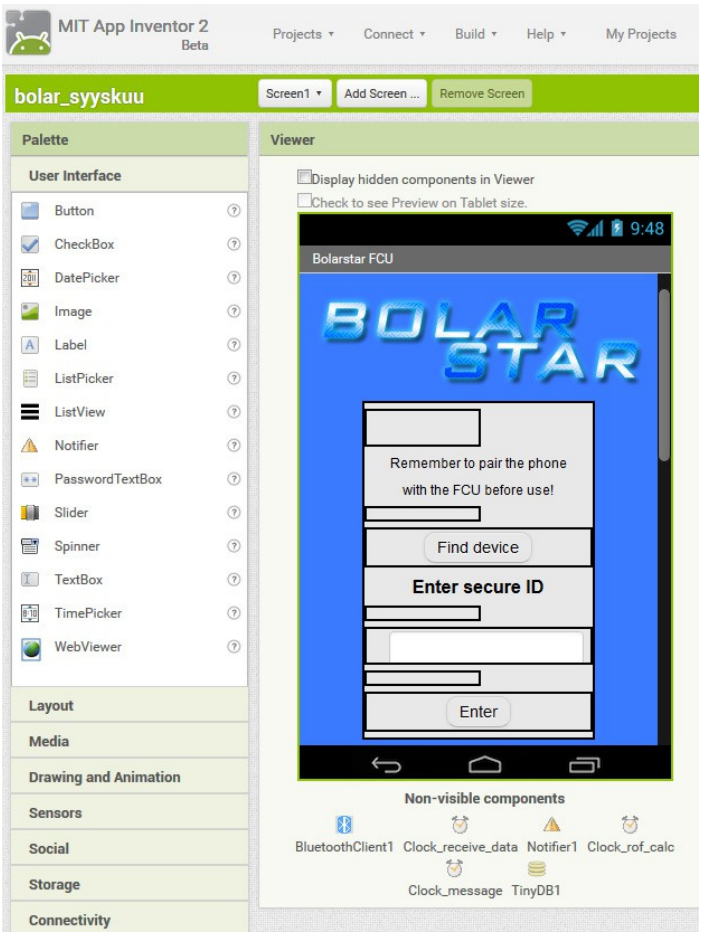
Ensimmäisessä Bluetooth-yhteensopivassa FCU-protyypissä käytettiin ilmaista Android älypuhelimen komentoriviohjelmaa, josta voi syöttää komentoja mille tahansa paritetulle Bluetooth-ohjaimelle (Kuva 2). Komentoriviohjelma toimi siten, että puhelin paritetaan normaalisti skannaamalla lähellä oleva Bluetooth-laitteita. FCU pyytää käyttäjää syöttämään paritusavaimen, minkä jälkeen suojattu yhteys on luotu. Kun yhteys on luotu, voidaan aukaista komentorivi-ohjelma. Tämä ratkaisu toimi hyvin, mutta ulkoasullisesti se oli hieman puutteellinen. Prototyypin testiohjelmaan pystyi syöttämään Bluetoothin kautta paineilmaventtiilille aukioloajan, kiinnioloajan millisekunteina ja pursketulioption. Kokeellisena arvona käytettiin ”soldenoid airflow %”-arvoa, jolla pystyi lähettämään 500Hz PWM-signaalia paineilmaventtiilille. Tästä arvosta ei lopulta ollut hyötyä, sillä venttiilin hystereesi ei sallinut vaihtelevaa syöttöjännitettä.



Kuva 2: Ensimmäinen toimiva Bluetooth-prototyyppi

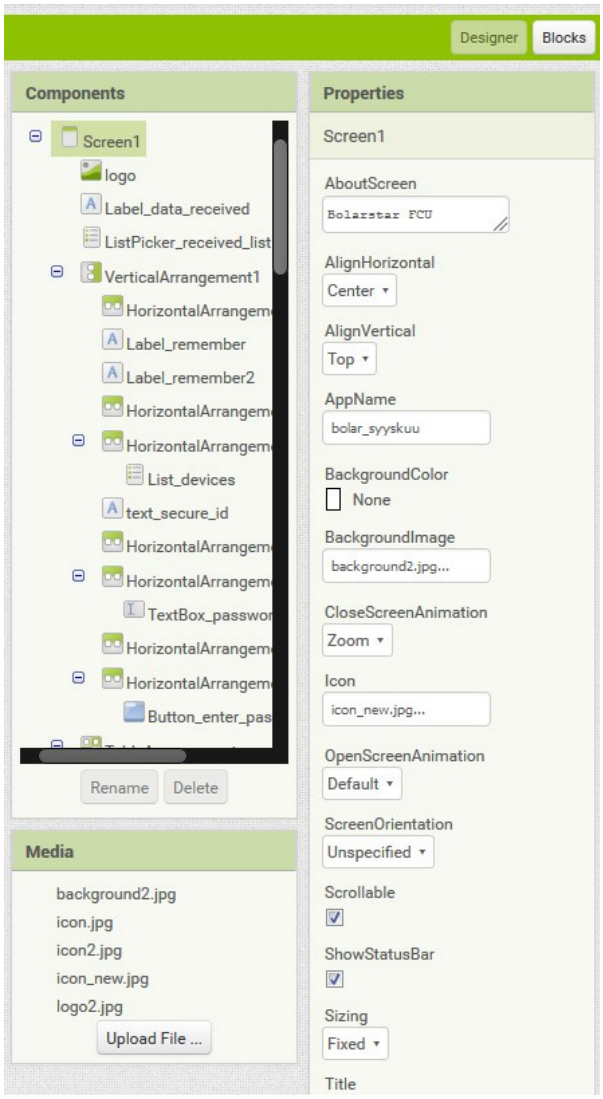
6.3 Mit App Inventor 2 Beta

Seuraava Bluetooth-ohjaimen Android-käyttöliittymäversio oli ohjelmoitu ilmaisella selainpohjaisella MIT App Inventor 2 beta:lla (Kuvio 11). Ohjelma on testattu toimivaksi 2010 julkaistussa Samsung I9000 Galaxy S Android-puhelimessa ja Android KitKat-versiossa. Ohjelmointi muistuttaa perinteistä if-else-ohjelmointia, mutta se tehdään raahaammalla palikoita piirtoalueelle (ks. liite 2). Sivulle kirjaudutaan Google-tunnuksilla, ja projektia voi jatkaa millä tietokoneella vaan, sillä projektit tallentuvat sivun tietokantaan.



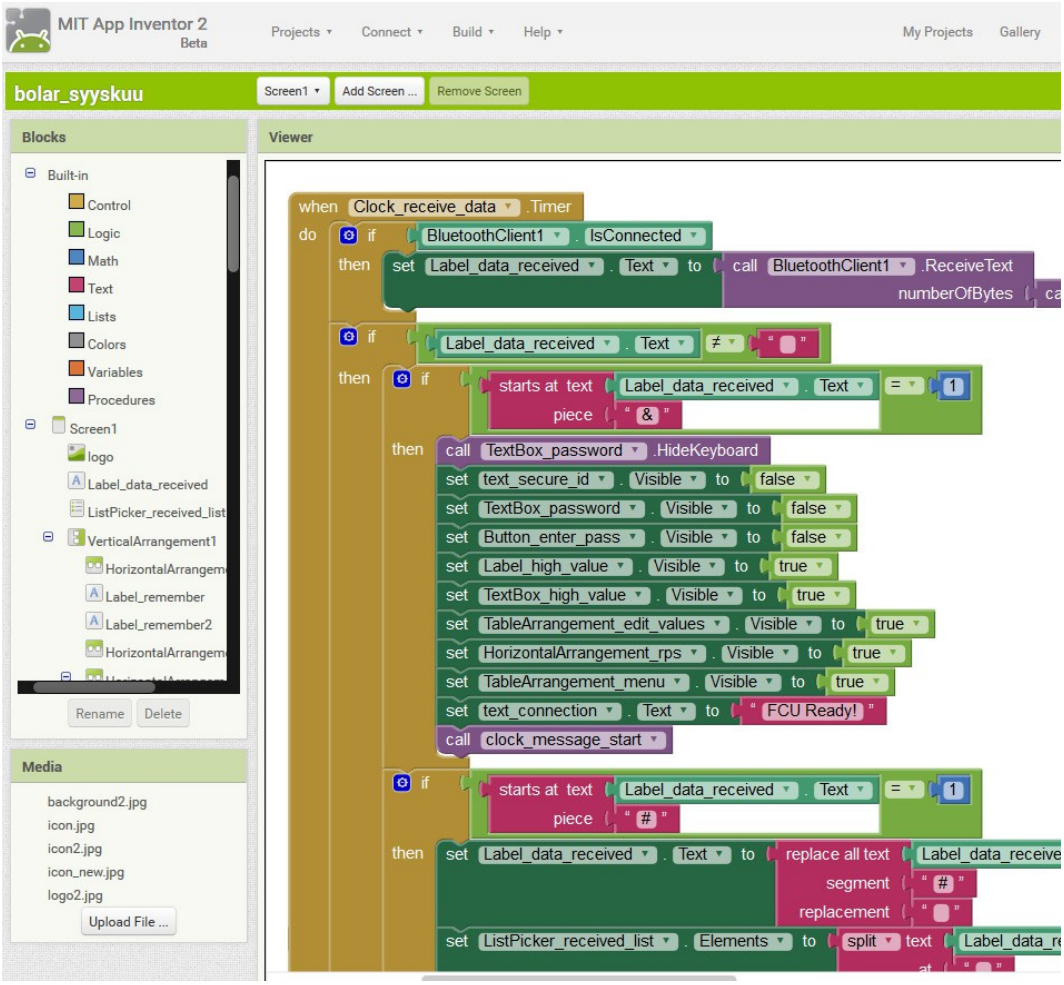
Kuvio 11: MIT App Inventorin työkalupalkki

Kun kirjaudutaan ensimmäistä kertaa sivulle, sinne tehdään uusi projekti klikkaamalla Projects-pudotusvalikkoa, minkä jälkeen valitaan ”start new project”. Projektin luomisen jälkeen päästään tekemään ohjelmaa. Vasemmalla näkyy MIT App inventorin työkalupalkki. Se sisältää kaikki ulkoasuun tarvittavat toiminnalliset objektit, jotka voidaan raahata vasen hiiren nappi pohjassa piirtoalueelle. Jotta käyttöliittymästä tulee responsiivinen, eli se mukautuu kaikkien laitteiden näytöille resoluutiosta riippumatta, täytyy ulkoasu suunnitella hyvin. Layout-valikosta löytyy taulukot, jolla voidaan pitää painikkeet ja tekstit paikoillaan suhteessa toisiinsa. Taulukoille voidaan antaa absoluuttisia arvoja, jolloin niiden koot ovat aina vakiot tai jättää leveys ja korkeus automaattiseksi, jolloin taulukko venyy näytön laidalta laidalle.



Kuvio 12: MIT App Inventorin komponenttivalikko

Piirtoalueen oikealta puolella on lista käytetyistä objekteista (Kuvio 12). Objekteille voi antaa erilaisia parametreja, fontteja, taustakuvia ja tekstiä. Siellä voidaan myös määrittää, miten puhelinsovellus käyttäytyy, kun näytön orientaatio muuttuu ja voiko näyttöä raahata ylös ja alas. Kun ohjelma käynnistetään, näytölle raahatut objektit eivät tee vielä mitään, koska niiltä puuttuu funktiot. Seuraavaksi lisätään objekteille funktioita painamalle ruudun oikeassa yläkulmassa olevaa blocks-painiketta.



Kuvio 13: MIT App Inventorin koodaus

Vasemmalla näkyvät valmis-funktiot, jolla hallitaan näytölle raahattuja objekteja (Kuvio 13). Ohjelmointi toimii siten, että klikataan jotakin valmisfunktiota ja raahataan sen piirtoalueelle. Ohjelmointi muistuttaa C-ohjelmointia, sillä se tulee monenlaisia silmukoita, taulukon käsittelyyn tarvittavia funktioita ja arvojen vertaamista. Syntaksin oppiminen on todella helppoa, sillä virheellistä palikkaa ei voi kiinnittää väärään palikkaan. Ohjelmointi ei vaadi laajaa ymmärrystä data-tyyppien muunnoksista, sillä ohjelma osaa tehdä muunnokset automaattisesti. Moni toiminto, joka saattaa olla monimutkainen tehdä puhtaalla C-kielellä, on yksinkertaistettu ohjelman käyttäjää varten. Tämä nopeuttaa ohjelman oppimista ja tekee ohjelman testauksesta helppoa.

Yllä olevassa esimerkissä ollaan 200 millisekunnin while-silmukassa, minkä jälkeen luetaan Bluetooth:in sarjaportista tullutta merkkijonoa. Jos

merkkijono sisältää #-merkin, merkkijono käsitellään ja arvot paristaan listaan.

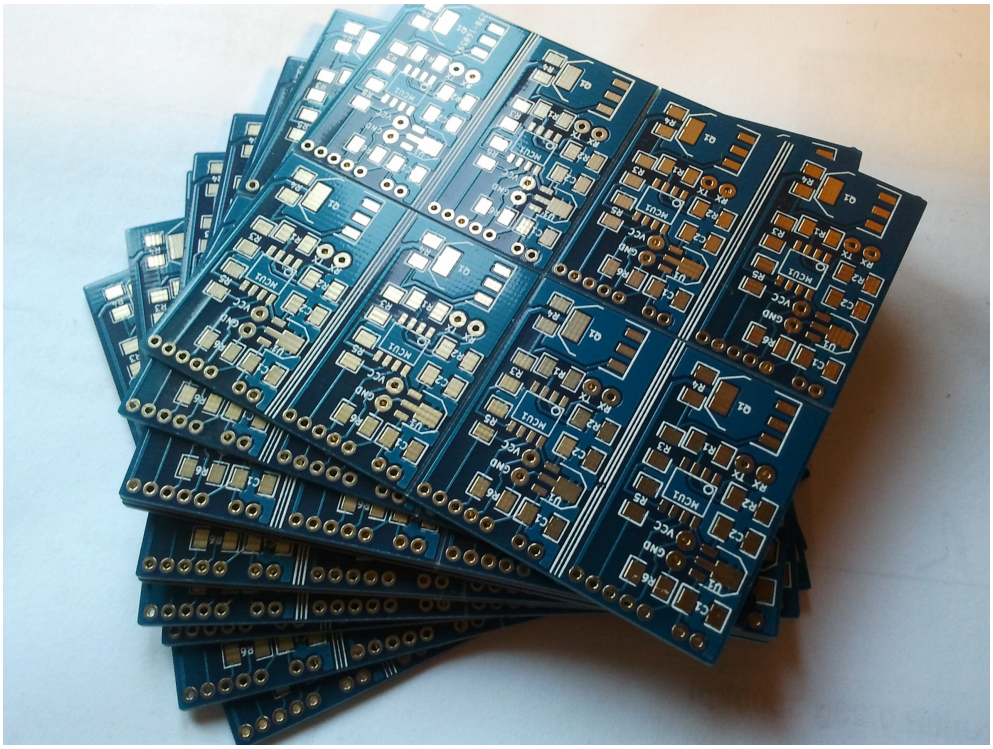
Ohjelman testaus tapahtuu generoimalla QR-koodi ohjelman sivulta. QR-koodi on linkki, jonka kautta puhelin hakee ohjelman internetistä. Ohjelman testaukseen tarvitaan MIT AI2 Companion -ohjelma. Toinen tapa testata ohjelmaa on luoda asennuspaketti ja lähettää se USB-kaapelia pitkin puhelimen muistiin. Asennuspaketin ajamiseen vaaditaan, että puhelin sallii kolmannen osapuolen ohjelmien asentamisen. Asennus tapahtuu muutamassa sekunnissa, ja vanha ohjelman versio ylikirjoitetaan uudella.

6.4 Valmis FCU-ohjain

Prototyyppien jälkeen kehitettiin uusi piirilevy, jonka tarkoitus oli helpottaa komponenttien kasaamista ja piirilevyllä toimivan ohjelman päivitystä. Lopullinen versio koostui printatusta PCB-levystä, pintaliitoskomponenteista ja liittimistä.

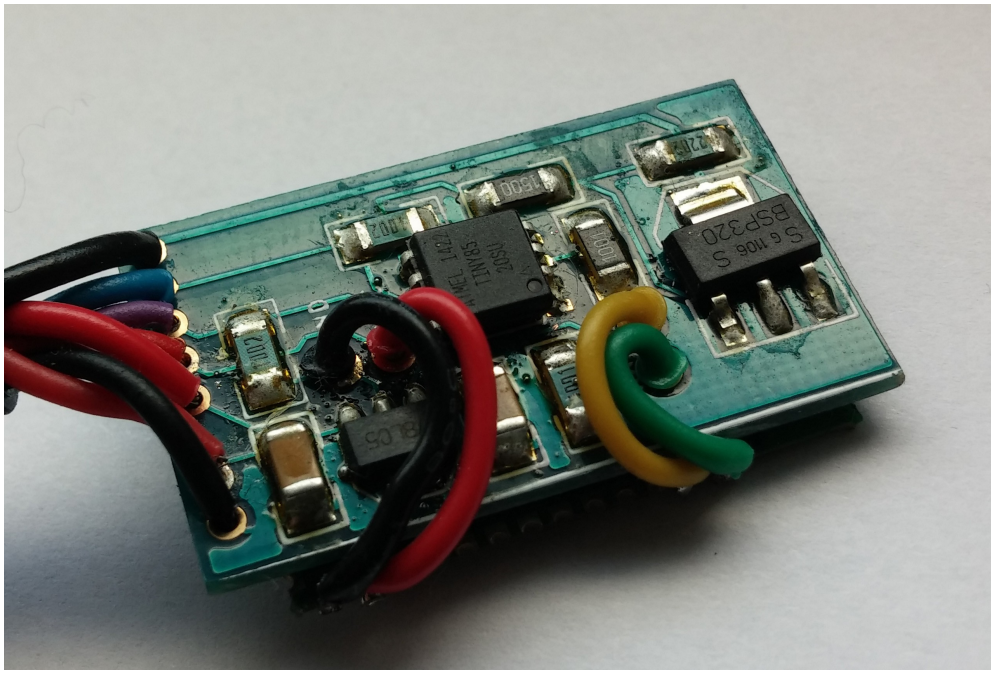
6.4.1 Valmis piirilevy

Prototyyppivaiheen jälkeen tarkoituksena oli kehittää lopullinen FCU, joka oli suunniteltu Ki-Cad suunnitteluohjelmalla. Kuvien piirtäminen alkoi piirikaavion piirtämisellä (ks. Liite 1), minkä jälkeen alkoi kaksikerroksisen PCB-levyn piirtäminen. Kun PCB-levyn gerber-kuvat ja porauskartat olivat valmiit (ks. Liite 3), ne lähtivät Aasiaan Itead-studios PCB-printtauspalveluun. Levyt tilattiin 1.0 mm -paksuisina ja ENIG-pinnoitteella. Piirilevyt saapuivat kuukauden odottelun jälkeen postiin noudettavaksi (Kuva 3).



Kuva 3: PCB-levyt

Piirilevy koostui 13 pintaliitoskomponentista, jotka testivaiheessa oli kolvattu kiinni käsin. Komponentit sekä piirilevyt ovat lyijyvapaita, jolloin valmis tuote ei riko RoHS-direktiiviä. Piirilevyn toiselle puolelle tuli kiinni HC-05 Bluetooth-ohjain, joka oli yhteydessä FCU:n mikrokontrolleriin kahdella hyppylangalla. Piirilevyn päässä on liittimet kahdelle mikrokytkimelle ja paineilmaventtiilille.



Kuva 4: Ensimmäinen valmis piirilevy

Kooltaan koko FCU komponentteineen oli 32x18x8 millimetriä (Kuva 4). Toimivan kokonaisuuden saaminen pieneen pakettiin oli haastava työ, joka vaati paljon uuden asian opettelua.

6.4.2 Mikrokontrollerin lähdekoodi

Mikrokontrollerin ohjelmointiin oli käytetty C-kieltä ja Arduino IDE-ohjelmaa, joka sisälsi valmiita kirjastoja ohjelmoinnin helpottamiseksi (kts. liite 4). Ohjelman tarkoitus on ohjata venttiilin toimintaan liittyvää ajastusta ja samanaikaisesti kuunnella serial-porttia. Jos serial-porttiin tulee dataa Bluetooth-moduulilta, data käsitellään normaalisti, minkä jälkeen venttiilin ohjaus voi jatkua.

Koodin rakennus alkoi siitä, että ensimmäisenä yritettiin lähettää viestiä mikrokontrollerilta puhelimeen. Ensimmäisessä prototyypissä oli hankaluuksia saada selkeäkielinen merkkijono Bluetooth-moduulin läpi puhelimen näytölle. Virheenä oli kuitenkin se, että merkkijonoa ei luettu vastaanottavassa päässä oikein. Tämä aiheutti omituisten merkkien ilmestymisen merkkijonon sijaan, mikä korjautui sillä, että serial portti konfiguroitiin uudestaan. Kun ensimmäinen merkkijono

saapui läpi virheettömänä, alkoi selvitys, miten merkkijonosta voisi poimia numeeriset arvot ja käyttää niitä hyödyksi. Näin alkoi yksinkertaisen protokollan kehitys, jonka avulla data siirretään laitteesta toiseen. Protokolla osaa tulkita kahta erilaista viestiä. Toinen on tarkoitettu lähettämään ainoastaan numeerisia arvoja ja toinen pelkän tekstipohjaisen viestin käyttäjälle. Tässä esimerkki protokollasta (Taulukko 2):

	dataprotokolla	esimerkkiarvot
Numeeristet arvot	#,A,B,C,D,E,F,G,H\n	#,0,123,7,20,3,0,0,30\n
teksti	@esimerkkiviesti\n	@FCU: New settings sent!\n

Taulikko 2: FCU-tiedonsiirtoprotokolla

Merkkien vastineet:

- #: Merkki, joka määrittää että viesti on päässyt perille. Jos merkkiä ei löydy viestin alusta, niin viesti suodatetaan pois.

- @: Merkki, joka tarkoittaa, että sen perässä on pelkästään tekstiä. Teksti on tarkoitettu näkymään käyttöliittymän näytöllä, jolloin siitä voi seurata mitä tiedonsiirrossa tapahtuu. Viesti päättyy rivinvaihtoon.

- A: Command-arvo määrittää, miten viestiä halutaan käsitellä. Normaalisti numeerisella viestillä halutaan lähettää uusia arvoja puhelimelta mikrokontrollerille, mutta sillä voidaan myös lähettää pyyntö salasanan vaihtoon ja sen hyväksymiselle, pyytää solenoidiventtiilin syklikertojen määrää numerona, pyytää arvoja mikrokontrollerilta lähetettäväksi puhelimeen ja lukituksen avaaminen, jolloin voidaan estää FCU:n väärinkäyttö. Command-arvoa ei lähetetä koskaan mikrokontrollerilta puhelimeen, sillä puhelin toimii aina master-laitteena ja mikrokontrolleri on aina slave.

- B: SecID on salainen numero, joka kulkee puhelimelta numeerisen viestin kanssa. Tällä voidaan estää väärinkäyttö, jos paritus-avain päätyy vääriin käsiin. Paritusavaimen vaihto vaatii Bluetooth-ohjaimen uudelleen konfiguroinnin AT-komennoilla. SecID-arvoa ei lähetetä koskaan mikrokontrollerilta puhelimeen, koska vastaanotettu oikea SecID on merkki siitä, että lähettävä laite turvallinen.

- C: High-value on venttiilin aukioloaika millisekunteina kertaa kymmenen. Koska reaalilukujen lähettäminen kokonaislukumuodossa osoittautui hankalaksi täytyy lukumuunnos tehdä vastaanottavassa päässä.

- D: Low-value on venttiilin kiinnioloaika millisekunteina kertaa kymmenen.

- E: Burst-size on pursketulioptio. Ohjelma toistaa sykliä arvon määrän verran jonka jälkeen sykli pysähtyy. Jos arvo on 0, niin sykli toistuu jatkuvasti liipaisinta painaessa.

- F: BoltActionDelay on arvo sekunteina. Se on viive, joka esiintyy jokaisen syklin jälkeen. Tämä voidaan asettaa nollasta useaan sekuntiin.

- G: ReloadDelay on arvo sekunteina. Se on viive, joka esiintyy seuraavan syklin jälkeen kun tietyn monta sylkiä on käyty läpi.

- H: MagSize on syklien määrä, joka viittaa arvoon G. MagSize määrittää kuinka monta sykliä voidaan ajaa ennen, kuin ReloadDelay-viive astuu voimaan.

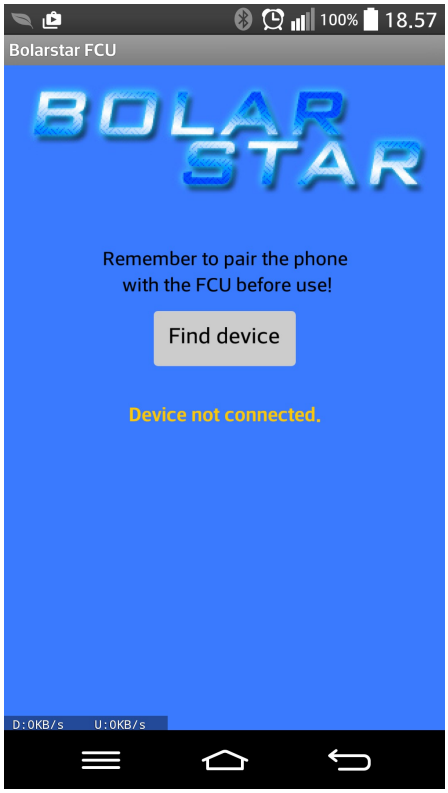
- \n: Tarkoittaa rivivaihtoa. Tämä kertoo serial portin puskurille, että viesti on valmis käsiteltäväksi. Ilman tätä merkkiä viesti jää puskuriin, ja viestin saa ulos ainoastaan lähettämällä porttiin rivinvaihdon uudelleen, mikäli se on kadonnut lähetysvaiheessa.

Protokollan käsittelyssä jouduttiin hyödyntämään merkkijonon parsimista, mikä tarkoittaa sitä, että vastaanotetut numeeriset arvot tallennetaan muistipaikoille. Parsimis-funktiolle täytyy kertoa, millä merkilllä arvot ovat erotettu, ja tässä käytettiin luonnollisesti pilkkua. Merkkijonon aloitusmerkki päättää, mihin tarkoitukseen viestiä käytetään. Ilman aloitusmerkkiä vastaanottava laite olettaa, että viesti on epäkelpo käsiteltäväksi. Tämä estää sen, että numeeriset muistipaikat eivät täyty virheellisellä datalla, mikä voi estää laitteen käytön täysin.

6.4.3 Valmis älypuhelinkäyttöliittymä

Uuden käyttöliittymän kehittämiseen oli käytetty samaa ohjelmaa kuin prototyypin tekemiseen, mutta siihen oli lisätty uusia ominaisuuksia ja uusi ulkoasu. Projekti kulki edelleen nimellä ”Bolarstar”, joka tulee vaihtumaan tulevaisuudessa.

Nykyinen sovellus sisälsi tunnusavaimen, joka vaadittiin laitteen parituksen jälkeen (Kuvio 14). Tämä esti muita ihmisiä tunkeutumasta laitteen asetuksiin mikäli paritusavain pääsi vääriin käsiin.



Kuvio 14: Uuden käyttöliittymän aloitusruutu

Kuvio 15: Varsinainen käyttöliittymä

Käyttöliittymä toimi siten, että numerokenttiin syötetään arvot, ja painetaan upload-painiketta (Kuvio 15). Painike lähettää arvot merkkijonona puhelimen Bluetooth-moduulin kautta radioaaltoina Bluetooth-vastaanotin moduulille. Moduuli lähettää arvot Attiny85-mikrokontrollerille, joka parsii arvot muistipaikkoihin ja mikrokontrollerin EEPROM Flash-muistiin. Painamalla download-painiketta käyttöliittymä lähettää käskyn mikrokontrollerille, jolloin se pyytää FCU:ta lähettämään nykyiset arvot käyttöliittymään. Arvot lähetetään mikrokontrollerin serial-portin kautta FCU:n Bluetooth-moduulille, josta merkkijono kulkee ilmateitse puhelimen Bluetooth-vastaanottimelle. Merkkijono parsitaan puhelimen ohjelmassa, josta ne syötetään tekstikenttiin loppukäyttäjää varten.

File-valikosta avautuu tallennusvaihtoehdot, jolloin asetukset voidaan tallentaa puhelimen muistiin ja ladata ne sieltä. Exit-painike sulkeen Bluetooth-yhteyden ja ohjelman.

7 YHTEENVETO

Työn tavoite oli tutkia Bluetooth:in sopivuutta automaatioympäristössä ja kehittää sille uusia sovelluskohteita. Vaikka Bluetooth toimisikin monessa laitteessa ongelmitta, langaton tiedonsiirto voi aiheuttaa vaihtelevaa tai katkonaista viivettä tiedon välityksessä. Bluetooth ei täten ole automaattisesti parempi valinta verrattuna perinteiseen kaapelointiin, koska kriittisen virheen sattuessa voi tapahtua kohtalokas tapaturma. Bluetooth:in käyttöä on tulevaisuudessa testattava enemmän ja enemmän, jotta se saadaan vastaamaan automaatioteollisuuden laatukriteerejä. Epävarma toiminta voi maksaa yritykselle paljon, jos langaton yhteys ei ole vakaa. Tämän takia on parempi ottaa Bluetooth-teknologian käyttöönotossa hitaita askelia ja käyttää vain uusimpia Bluetooth-yhteensopivia komponentteja.

Tietoturvariskit tuovat omat ongelmat langattomaan tiedonsiirtoon. Bluetooth ei kuitenkaan ole täysin varma systeemi sillä sitä rajoittaa PIN-koodin lyhyt merkkimäärä, joka on mahdollista murtaa auki. Langatonta viestintää voi kuunnella kuka vaan, jos paritusavain on päässyt vääriin käsiin. Vanhemmat Bluetooth-versiot ovat kaikista alttiimpia tietomurrolle, mutta tietoturvaa kehitetään jatkuvasti eteenpäin Bluetooth SIG:n toimesta.

Opinnäytetyön käytännöntyöhön kuului langattoman Bluetooth-paineilmaventtiiliohjaimen suunnittelu ja rakennus. Työssä päästiin tavoitteeseen, jolloin mikrokontrolleri saatiin kommunikoimaan älypuhelimen kanssa ja hyödyntää lähetettyä dataa parametrien määrittämisessä. Työn hankalin vaihe oli uuden asian opettelu, mutta kaiken tämän jälkeen opittuja taitoja voidaan nyt soveltaa monimutkikkaampien laitteiden ohjaamiseen. Ilmaiset ohjelmat mahdollistavat kaikille kiinnostuneille Bluetooth:in opettelun, ja tarvittavien komponenttien hinnat eivät ole kovin suuret. Bluetooth:ia voi soveltaa esimerkiksi kotona ohjaamaan automaattisia laitteita.

8 LÄHTEET

Grandlund, K (2001.) Langaton tiedonsiirto: Langattoman tiedonsiirron peruskirja 288 s.

Bluetooth SIG, Inc (2016a). Basic Rate/Enhanced Data Rate (BR/EDR) [viitattu 9.5]. Saatavissa: https://developer.bluetooth.org/TechnologyOverview/Pages/BREDR.aspx

Bluetooth SIG, Inc (2016b). Bluetooth high speed [viitattu 9.5]. Saatavissa: https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-technology-basics/bluetooth-high-speed

Heydon, R (2013a). Basics & Brand | Ch. 4: Low Energy [viitattu 9.5]. Saatavissa: https://www.youtube.com/watch?v=BCSpfxhIJGk

RF Wireless World (2012). Bluetooth Power classes-class1,class2,class3 [viitattu 9.5]. Saatavissa: http://www.rfwireless-world.com/Tutorials/Bluetooth-power-classes.html

Hodgdon, C (2003). Adaptive Frequency Hopping for Reduced Interference between Bluetooth® and Wireless LAN [viitattu 9.5]. Saatavissa: http://www.design-reuse.com/articles/5715/adaptive-frequency-hopping-for-reduced-interference-between-bluetooth-and-wireless-lan.html

Gelzayd, Y (2002). AN ALTERNATE CONNECTION ESTABLISHMENT SCHEME IN THE
BLUETOOTH SYSTEM [viitattu 9.5]. Saatavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.548.9978&rep=rep1&type=pdf

Galeev, M (2011). Bluetooth 4.0: An introduction to Bluetooth Low Energy —Part II [viitattu 9.5]. Saatavissa: http://www.eetimes.com/document.asp?doc_id=1278966

Liu, S (2016). Bluetooth Overview [viitattu 9.5]. Saatavissa: http://progtutorials.tripod.com/Bluetooth_Technology.htm

Mäkelä, M (2006a). USE OF BLUETOOTH IN AUTOMATION TECHNOLOGY 28 s. [viitattu 9.5]. Saatavissa: https://www.theseus.fi/handle/10024/1059

Bluetooth SIG, Inc (2016c). Core System Architecture [viitattu 9.5]. Saatavissa: https://developer.bluetooth.org/TechnologyOverview/Pages/Core.aspx

Bluetooth SIG, Inc (2016d). Host Controller Interface (HCI) Architecture [viitattu 9.5]. Saatavissa: https://developer.bluetooth.org/TechnologyOverview/Pages/HCI.aspx

IEEE, NPO (1999). Specification of the Bluetooth System 524 s. [viitattu 9.5]. Saatavissa: http://grouper.ieee.org/groups/802/15/Bluetooth/core_10_b.pdf

Padgette, J; Scarfone, K; Chen, L (2012). Guide to Bluetooth Security [viitattu 9.5]. Saatavissa: http://csrc.nist.gov/publications/nistpubs/800-121-rev1/sp800-121_rev1.pdf

FINLEX (2007). Laki rikoslain muuttamisesta [viitattu 9.5]. Saatavissa: http://www.finlex.fi/fi/laki/alkup/2007/20070540#Lidm44496

Haataja, K; Hyppönen, K; Pasanen, S; Toivanen, P (2013). Bluetooth Security Attacks, Comparative Analysis, Attacks, and Countermeasures [viitattu 9.5] Saatavissa: http://www.springer.com/gp/book/9783642406454

Bluetooth SIG, Inc (2016e). Security, Bluetooth Smart (Low Energy) [viitattu 9.5]. Saatavissa: https://developer.bluetooth.org/TechnologyOverview/Pages/LE-Security.aspx

Pushpa, R (2007). Bluetooth network-the adhoc network concept [viitattu 9.5]. Saatavissa: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4147524

Heydon, R (2013b). Basics & Brand | Ch. 3: Basic Rate/Enhanced Data Rate/High Speed [viitattu 9.5]. Saatavissa: https://www.youtube.com/watch?v=XmgpQKfvmYI

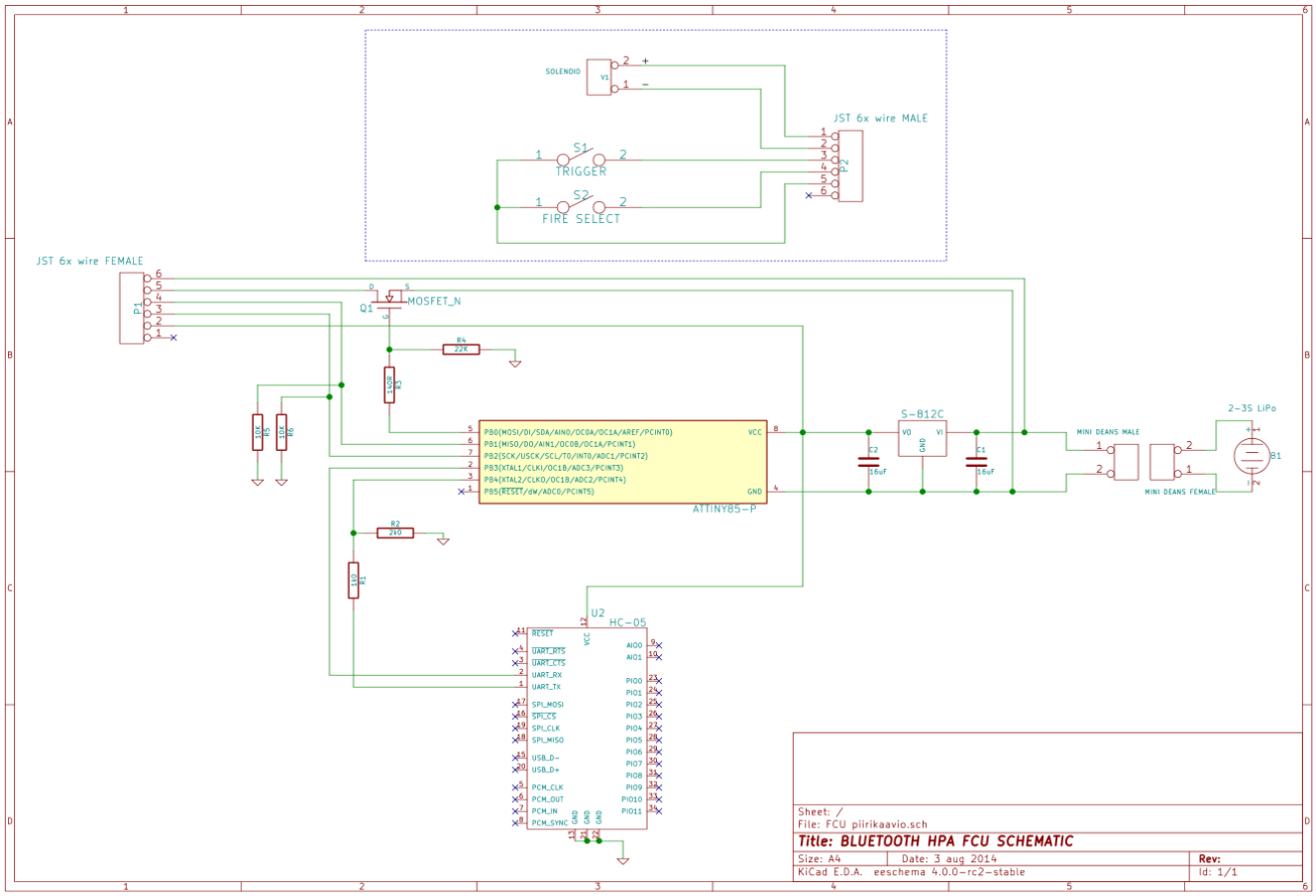
Heydon, R (2013c). Low Energy | Ch. 1 - Technology Features [viitattu 9.5]. Saatavissa: https://www.youtube.com/watch?v=iZSzqFF7rCs

Phoenix Contact, Ltd (2016). Bluetooth I/O [viitattu 9.5]. Saatavissa: https://www.phoenixcontact.com/online/portal/us?1dmy&urile=wcm:path:/usen/web/main/products/subcategory_pages/bluetooth_io_p-26-03-02/542d2f68-9d3b-46f3-b82c-4550965b52cc

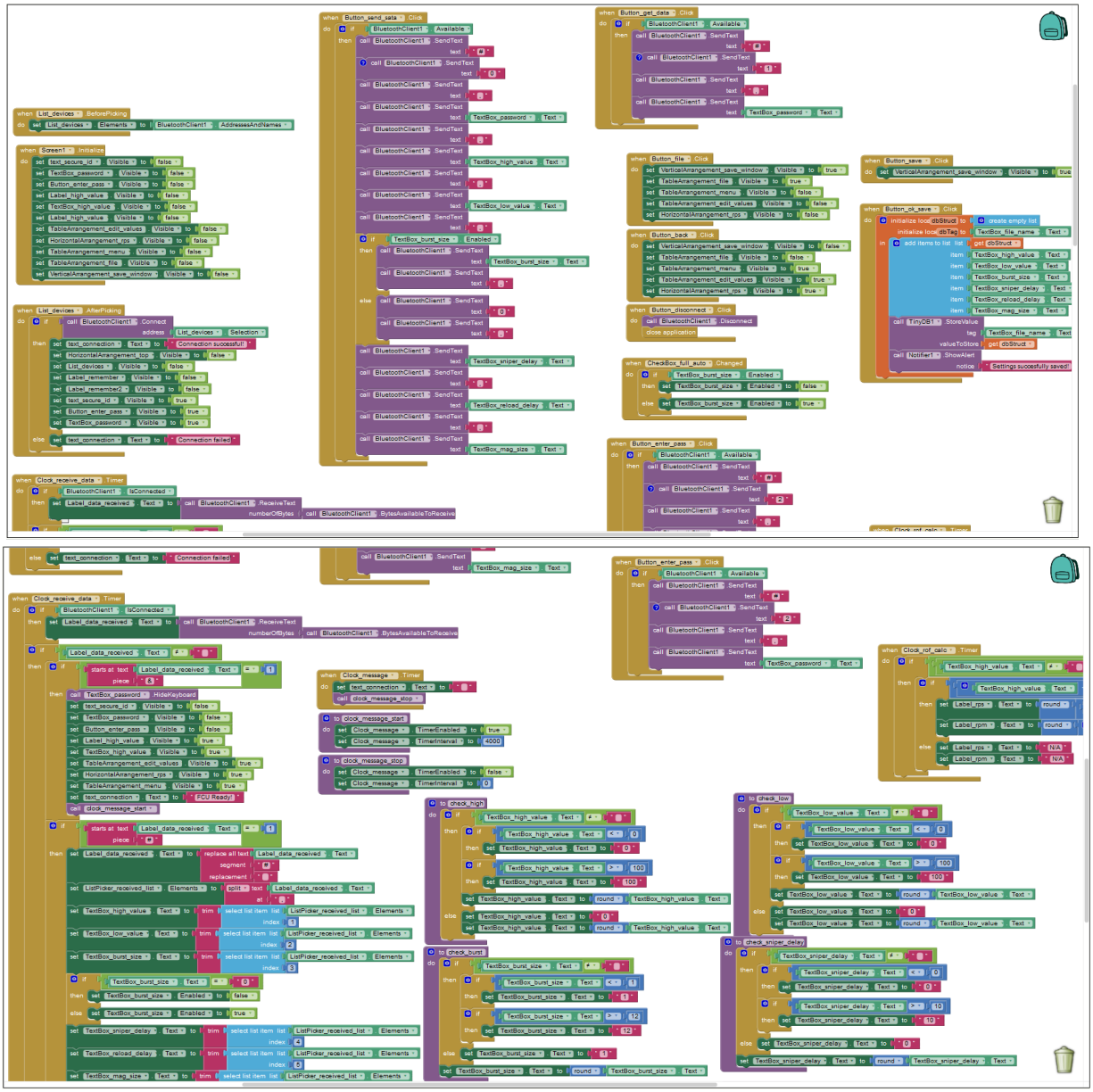
Seltec Automation LLP (2015). Phoenix Contact Industrial Bluetooth [viitattu 9.5]. Saatavissa: http://www.seltec.co.uk/products/phoenix-contact-industrial-bluetooth.html

9 LIITTEET

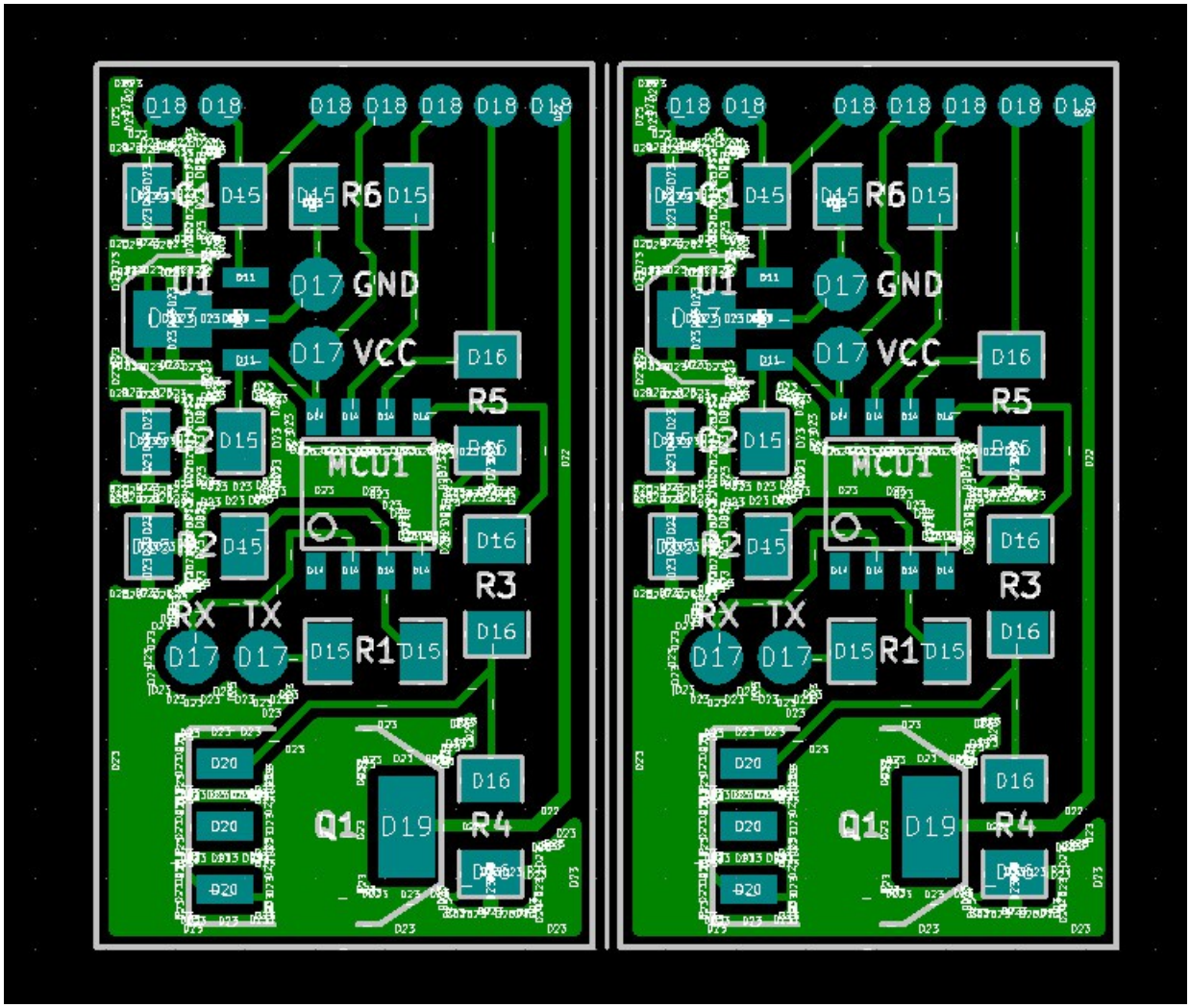
Liite 1: FCU-piirilevyn sähkökuva.



Liite 2: Esimerkki Bluetooth-käyttöliittymän koodista.



Liite 3: FCU-piirilevyn gerber-kuvat graafisessa muodossa.



Liite 4: FCU-mikrokontrollerin lähdekoodi.

```
/*
BUGIT:

*/

/*
DATAPROTOKOLLA

FCU->SMARTPHONE
SETTINGS: #,A,B,C,D,E
MESSAGE: @Hello World!

SMARTPHONE->FCU
SETTINGS: #,A,B,C,D,E
MESSAGE: @Hello World!

# = START CHAR
A = COMMAND
  0 = CHANGE SETTINGS
  1 = SEND SETTINGS
  2 = PASS OK
B = HIGH

C = LOW
D = BURSTSIZE (0=AUTO)

*/

#include <EEPROM.h>
#include <SoftwareSerial.h>

#define trigger 1 // Trigger pin, pulldown
#define full 2 // Full-auto pin, pulldown
#define ledPin 0 // goes to an fet to turn on the led & solenoid, solenoidi
pulldown 10k
#define txPin 4
#define rxPin 3

//FIRING SETTINGS
int secId = 123;
int highValue = 25;
int lowValue = 25;
int burstSize = 3;
int boltActionDelay = 0;
int reloadDelay = 0;
int magSize = 0;
int antiTheft = 0;

int burstCount = 0;
int gunLocked = 0;
int magNow = 0;
int shots = 0;
boolean triggerDown = false;
String txt = "";
unsigned long shotCounter = 0;

SoftwareSerial mySerial(rxPin, txPin); // RX, TX

void firstRun()
{
  if (EEPROM.read(255) == 255)
  {
    //WRITE DEFAULTS
    EEPROM.write(1, secId);
    EEPROM.write(2, highValue);
    EEPROM.write(3, lowValue);
    EEPROM.write(4, burstSize);
    EEPROM.write(5, boltActionDelay);
    EEPROM.write(6, reloadDelay);
    EEPROM.write(7, magSize);
```

```
    EEPROM.write(8, antiTheft);
    EEPROM.write(255, 0);
  }
}

void setup()
{
  //default settings
  firstRun();

  secId = EEPROM.read(1);
  highValue = EEPROM.read(2);
  lowValue = EEPROM.read(3);
  burstSize = EEPROM.read(4);
  boltActionDelay = EEPROM.read(5);
  reloadDelay = EEPROM.read(6);
  magSize = EEPROM.read(7);
  magNow = magSize;
  antiTheft = EEPROM.read(8);

  if (antiTheft != 0)
  {
    gunLocked = 1;
  }

  pinMode(trigger,INPUT);
  pinMode(full,INPUT);
  pinMode(ledPin,OUTPUT);
  pinMode(txPin,OUTPUT);
  pinMode(rxPin,INPUT);
  mySerial.begin(9600);
  mySerial.flush();
}

void outputHIGH()
{
  digitalWrite(ledPin,HIGH);

  if (highValue>0)
  {
    delay(highValue);
  }
}

void outputLOW()
{
  digitalWrite(ledPin,LOW);

  if (lowValue>0)
  {
    delay(lowValue);
  }
}

void fire()
{
  if (gunLocked == 0)
  {
    outputHIGH();
    outputLOW();
    magNow-=1;
    shotCounter+=1;
  }
}

void loop()
{
  int readTrigger = digitalRead(trigger); //8
  int readFull = digitalRead(full); //7
  int fautoEnabled = false;

  if (readTrigger == HIGH && readFull == HIGH && burstSize == 0)
  {
    fautoEnabled = true;
```

```
}

//burst fire
if (readTrigger == HIGH)
{
  if (triggerDown == false)
  {
    if (readFull == LOW)
    {
      //SEMI
      shots+=1;
    }
    else
    {
      //BURST
      shots+=burstSize;
    }
    triggerDown = true;
  }
}
else
{
  triggerDown = false;
}

//HOITAA LAUKAUKSET
if (shots>0 || fautoEnabled == true)
{
  //JOS FAKE RELOAD TRUE
  int magEnabled = false;
  if (magSize > 0 && reloadDelay > 0){
    magEnabled = true;
  }

  //JOS FULL AUTO
  if (fautoEnabled == true)
  {
    fire();
  }

  //JOS BURST
  while(shots>0)
  {
    fire();
    shots-=1;

    //JOS LIPAS LOPPUU KESKEN BURSTIN
    if (magNow == 0 && magEnabled == true)
    {
      shots = 0;
    }
  }

  //just in case
  if (shots < 0)
  {
    shots = 0;
  }

  //control fake reload
  if (magNow <= 0 && magEnabled == true)
  {
    delay(reloadDelay*1000);
    magNow = magSize;
  }

  //sniper delay
  if (boltActionDelay > 0)
  {
    delay(boltActionDelay*1000);
  }
}
```

```
char get_char = ' ';

//wait for incoming data
if (mySerial.available() < 1) return; // if serial empty, return to loop().

//parse start flag
get_char = mySerial.read();
if (get_char != '#') return; // if no command start flag, return to loop().

//mySerial.println("@"+get_char);

int command = mySerial.parseInt();//COMMAND
int var1 = mySerial.parseInt();//secId
int var2 = mySerial.parseInt();//highValue
int var3 = mySerial.parseInt();//lowValue
int var4 = mySerial.parseInt();//burstSize
int var5 = mySerial.parseInt();//boltActionDelay
int var6 = mySerial.parseInt();//reloadDelay
int var7 = mySerial.parseInt();//magSize

//# = parse command
//@ = text command

if (secId == var1)
{
  //GET: SETTINGS
  if (command == 0)
  {
      highValue = var2;
      lowValue = var3;
      burstSize = var4;
      boltActionDelay = var5;
      reloadDelay = var6;
      magSize = var7;
      magNow = magSize;

      EEPROM.write(2, highValue);
      EEPROM.write(3, lowValue);
      EEPROM.write(4, burstSize);
      EEPROM.write(5, boltActionDelay);
      EEPROM.write(6, reloadDelay);
      EEPROM.write(7, magSize);
      mySerial.print("@FCU: New settings received!\n");
  }
  //SEND: SETTINGS
  if (command == 1)
  {
    mySerial.print("#");
    mySerial.print(highValue);
    mySerial.print(",");
    mySerial.print(lowValue);
    mySerial.print(",");
    mySerial.print(burstSize);
    mySerial.print(",");
    mySerial.print(boltActionDelay);
    mySerial.print(",");
    mySerial.print(reloadDelay);
    mySerial.print(",");
    mySerial.print(magSize);
    mySerial.print(",");

    mySerial.print("\n");
    delay(1000);
    mySerial.print("@FCU: New settings sent!\n");
  }

  //SEND: PASSWORD CONFIRMATION
  if (command == 2)
  {
    mySerial.println("&asd");
    //mySerial.println("@Correct pass! fcu");
  }

  //GET: PASSWORD CHANGE
```

```
    if (command == 3)
    {
      //new password
      if (var2 > 0)
      {
        if (var2 < 256)
        {
          secId = var2;
          EEPROM.write(1, secId);
          mySerial.print("@Done! The new Sec ID is: ");
          mySerial.print(secId);
          mySerial.print("\n");
        }
        else
        mySerial.print("@Error! The number must be between 0-255.\n");
      }
      else
      mySerial.print("@Error! The number must be between 0-255.\n");
    }

    //SEND: shot counter
    if (command == 4)
    {
      mySerial.print("@FCU: Shots fired: ");
      mySerial.print(shotCounter);
      mySerial.print("\n");
      //mySerial.println("@Correct pass! fcu");
    }
    //GET: ANTI-THEFT
    if (command == 5)
    {
      if (var2 == 0)
      {
        EEPROM.write(8, 0);
        mySerial.print("@FCU: Anti-Theft disabled!\n");
      }
      else
      {
        EEPROM.write(8, 1);
        mySerial.print("@FCU: Anti-Theft enabled!\n");
      }
    }
    //GET: UNLOCK GUN
    if (command == 6)
    {
      if (var2 == 0)
      {
        gunLocked = 0;
        mySerial.print("@FCU: FCU unlocked!\n");
      }
    }
  }
  //SEND: ERROR REPORT
  else
  {
    mySerial.println("@Incorrect ID!");
  }

  mySerial.flush();
}
```